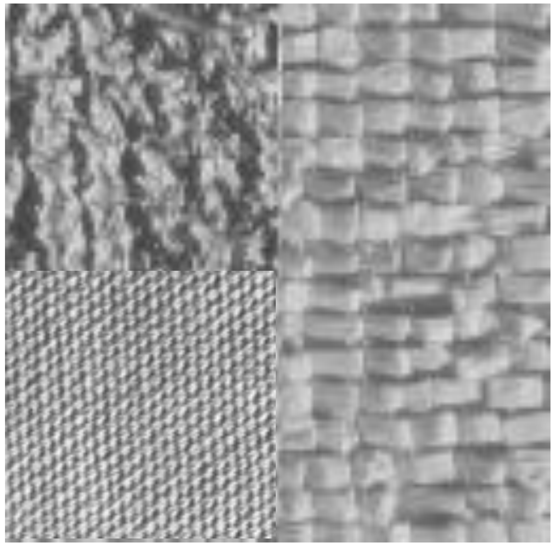
Image segmentation

GW Chapter 10

Pratt Chapter 7

Image segmentation

- Goal: partition the image into its constituent objects
- Approaches
  - Discontinuity: detect abrupt chsnges in gray levels ➔ edge detection
  - Similarity: group pixels based on their similarity with respect to a predefined criterion ➔ region-based processing
    - Feature extraction
    - Region growing
    - Feature clustering/classification



Edge detection

Digital Image Processing

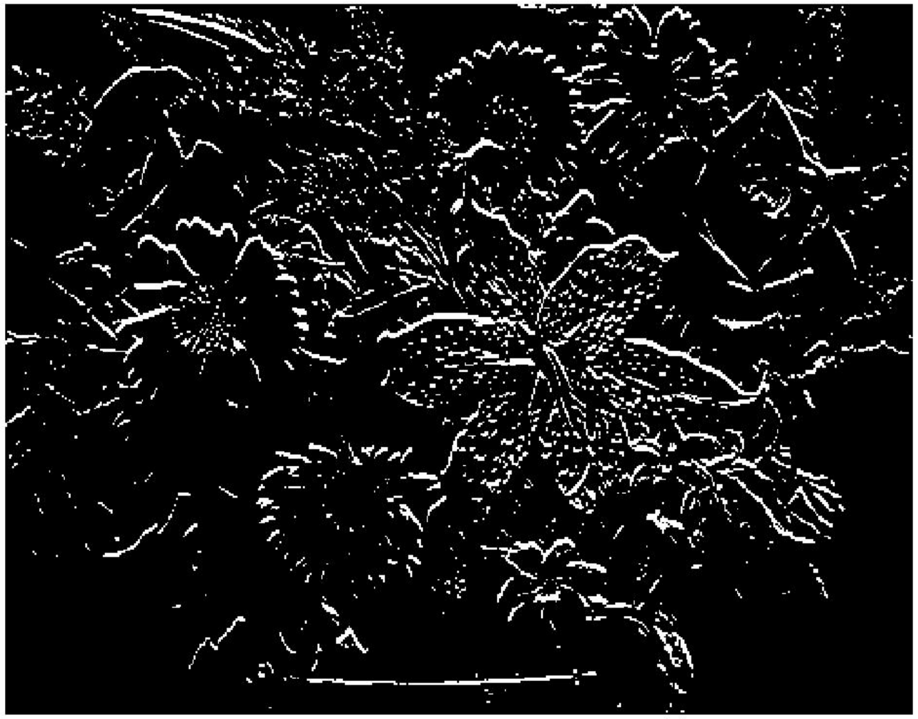
K. Pratt, Chapter 15, pag 443

GW Chapter 10

Edge detection

- Framework: image segmentation
- Goal: identify objects in images
  - but also feature extraction, multiscale analysis, 3D reconstruction, motion recognition, image restoration, registration
- Classical definition of the edge detection problem: localization of *large local changes* in the grey level image → large graylevel *gradients*
  - This definition does not apply to *apparent edges*, which require a more complex definition
  - Extension to color images
- Contours are very important perceptual cues!
  - They provide a first *saliency map* for the interpretation of image semantics

Contours as perceptual cues



Contours as perceptual cues

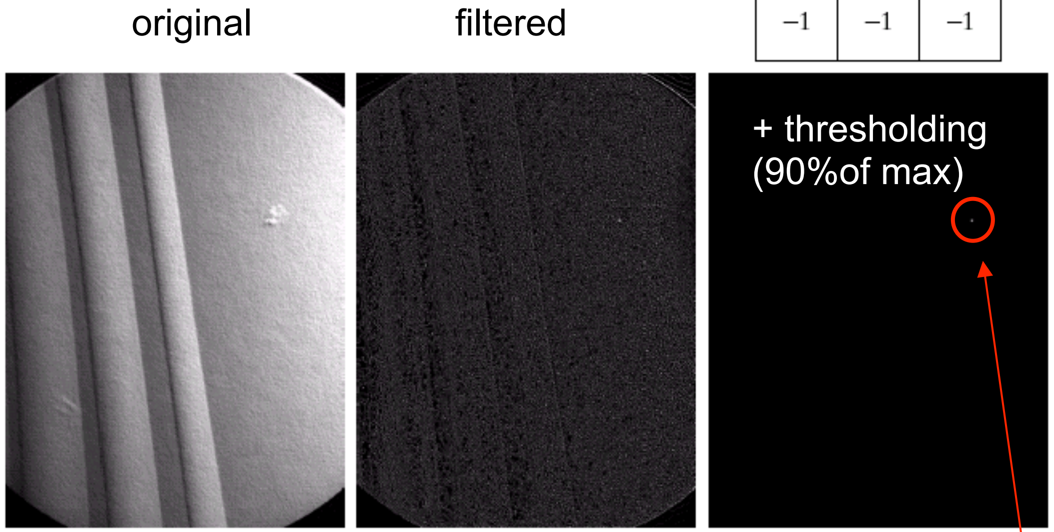
What do we detect?

- Depending on the impulse response of the filter, we can detect different types of graylevel discontinuities
  - Isolate points (pixels)
  - Lines with a predefined slope
  - Generic contours
- However, edge detection implies the evaluation of the local gradient and corresponds to a (directional) derivative

Detection of Discontinuities

- Point Detection

–1	–1	–1
–1	8	–1
–1	–1	–1



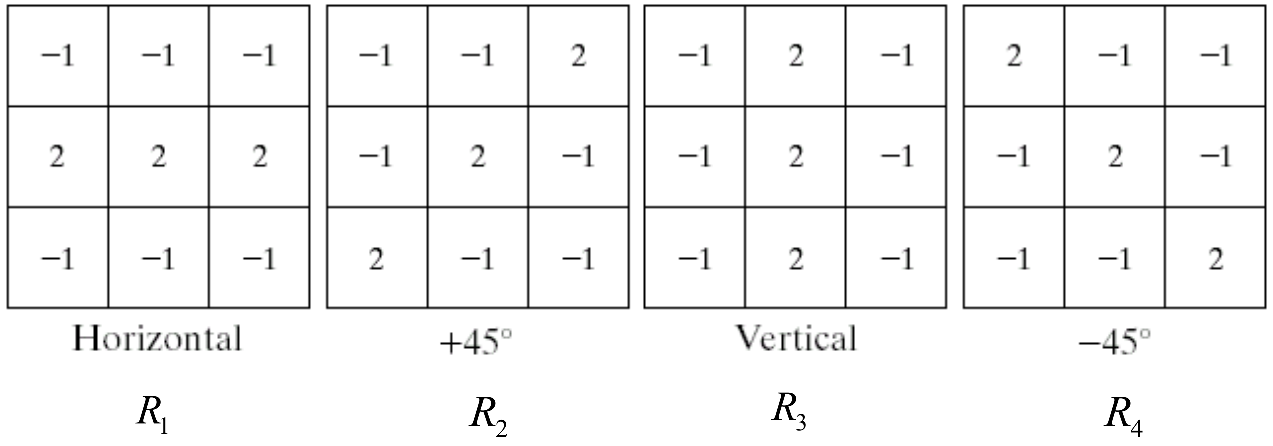
a
b c d

FIGURE 10.2 (a) Point detection mask. (b) X-ray image of a turbine blade with a porosity. (c) Result of point detection. (d) Result of using Eq. (10.1-2). (Original image courtesy of X-TEK Systems Ltd.)

Detection of Discontinuities

- Line detection

FIGURE 10.3 Line masks.



−1	−1	−1
2	2	2
−1	−1	−1

Horizontal

R_1

−1	−1	2
−1	2	−1
2	−1	−1

+45°

R_2

−1	2	−1
−1	2	−1
−1	2	−1

Vertical

R_3

2	−1	−1
−1	2	−1
−1	−1	2

−45°

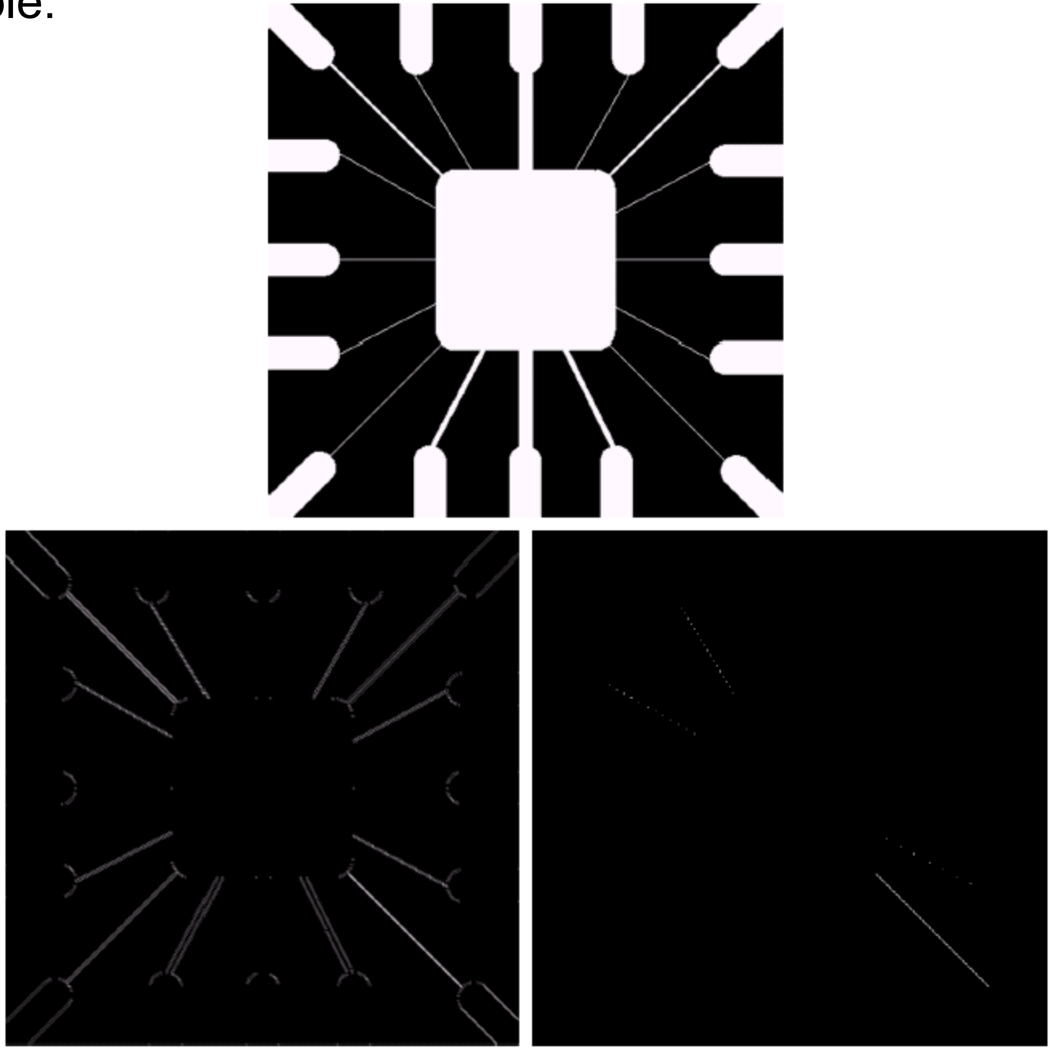
R_4

Detection of Discontinuities

- Line Detection Example:

a
b c

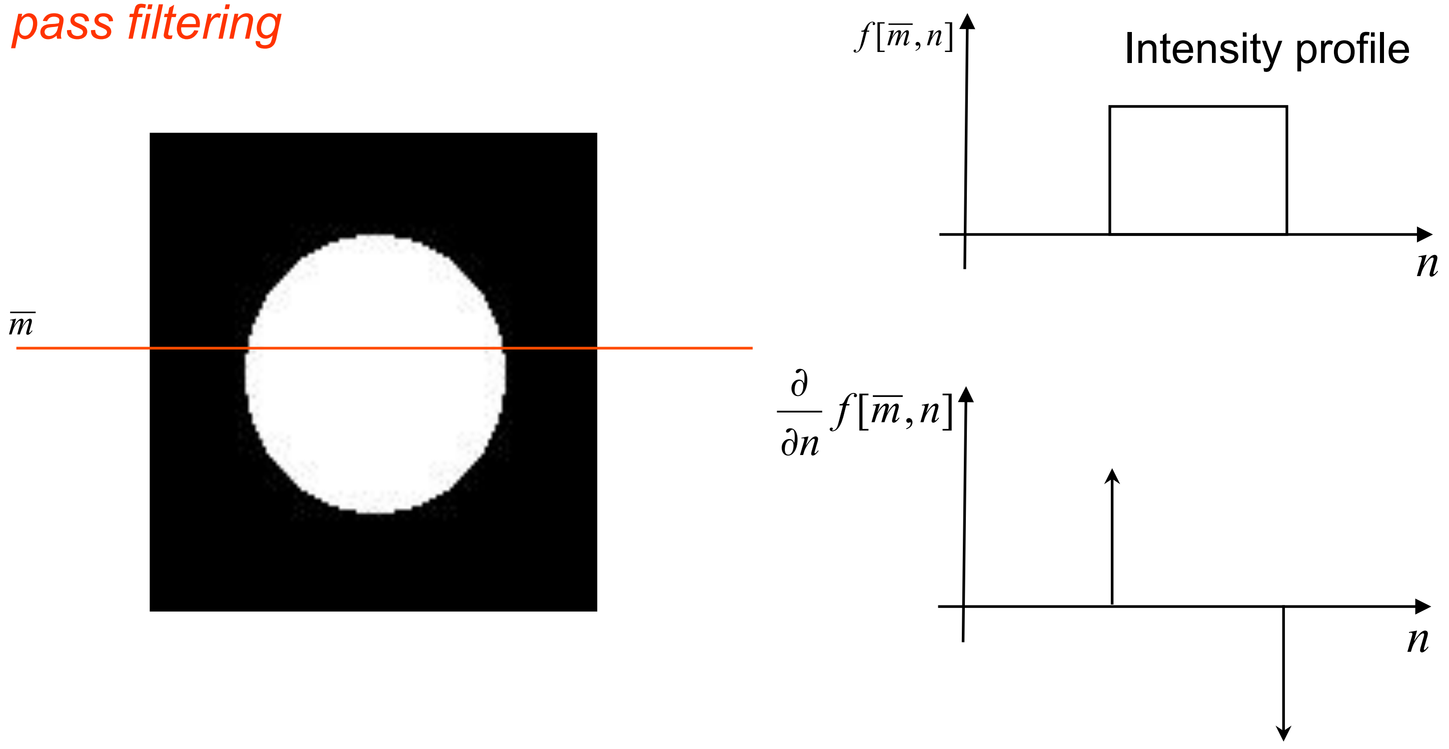
FIGURE 10.4 Illustration of line detection. (a) Binary wire-bond mask. (b) Absolute value of result after processing with −45° line detector. (c) Result of thresholding image (b).



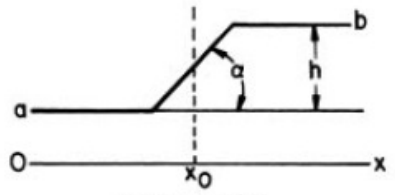
Threshold=max{filtered}
Suitable for binary images

Edge detection

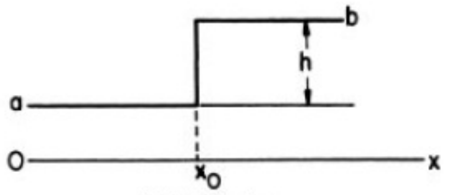
- Image locations with abrupt changes → *differentiation → high pass filtering*



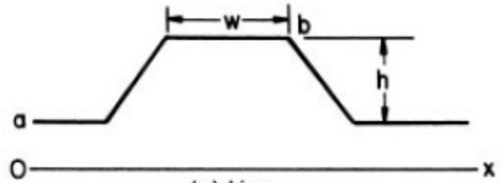
Types of edges



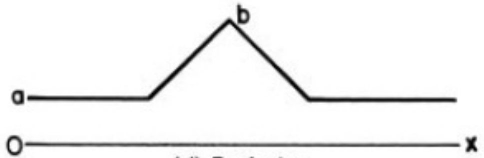
(a) Ramp edge



(b) Step edge

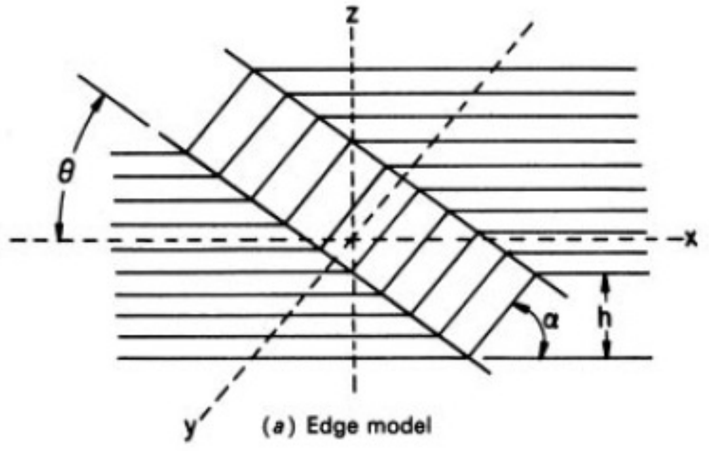


(c) Line

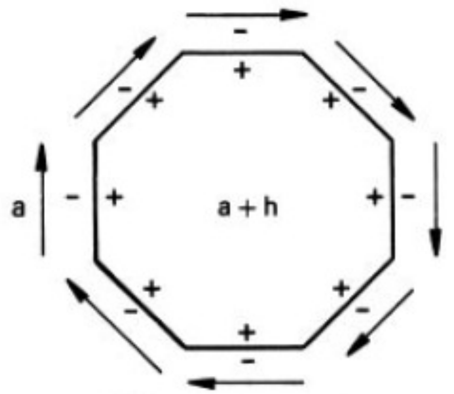


(d) Roof edge

Continuous domain edge models



(a) Edge model



(b) Edge direction definition

2D discrete domain single pixel spot models

a	a	a	a	a	a	a
a	a	a	a	a	a	a
a	a	a	a	a	a	a
a	a	a	b	a	a	a
a	a	a	a	a	a	a
a	a	a	a	a	a	a
a	a	a	a	a	a	a

Step spot

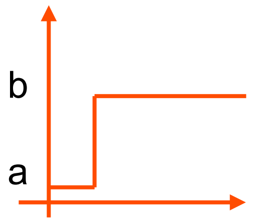
a	a	a	a	a	a	a
a	a	a	a	a	a	a
a	a	c	c	c	a	a
a	a	c	b	c	a	a
a	a	c	c	c	a	a
a	a	a	a	a	a	a
a	a	a	a	a	a	a

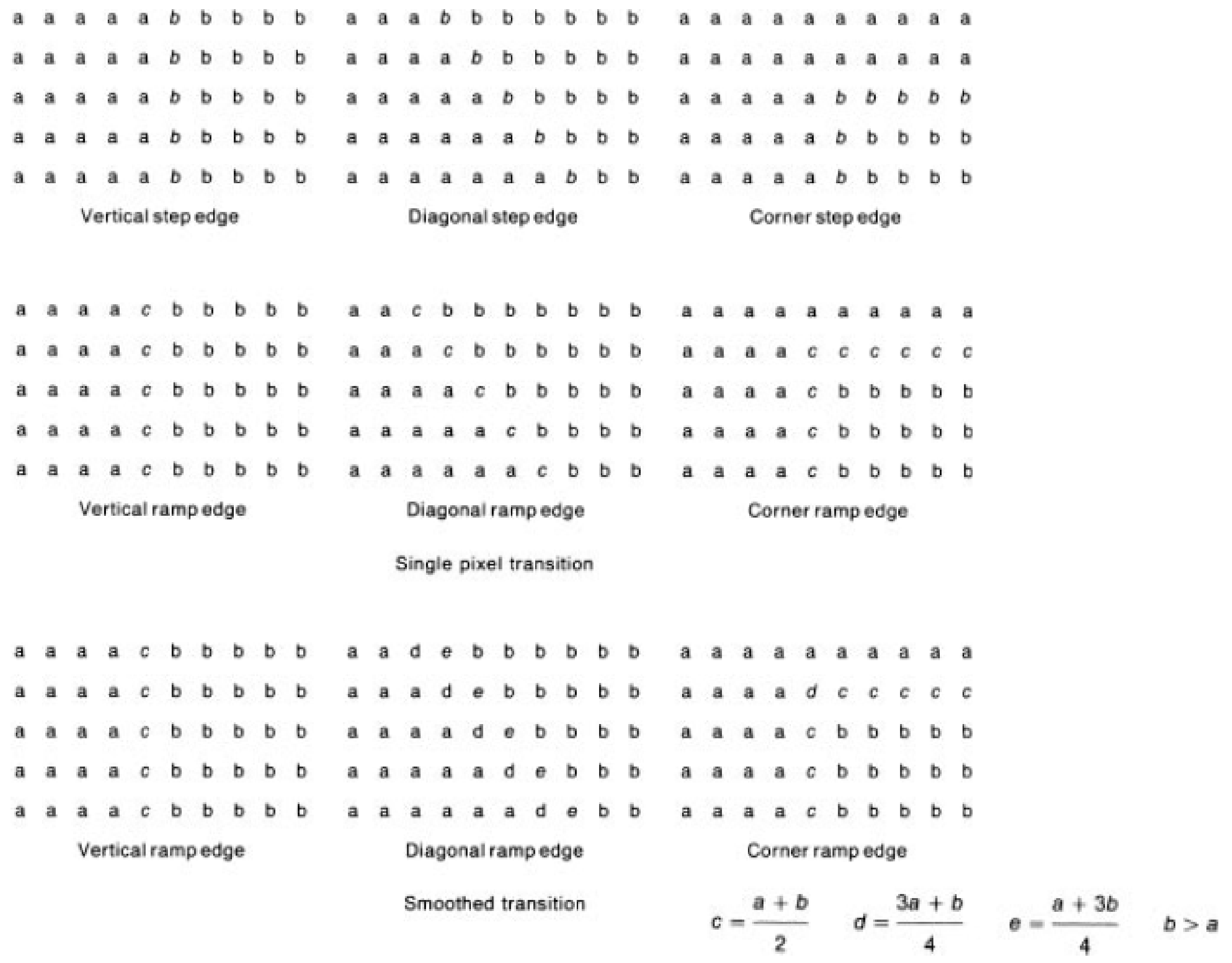
Single pixel transition spot

a	a	a	a	a	a	a
a	a	a	a	a	a	a
a	a	a	a	a	a	a
a	a	a	d	d	a	a
a	a	a	d	d	a	a
a	a	a	a	a	a	a
a	a	a	a	a	a	a

Smoothed transition spot

Discrete domain edge models





a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b

Vertical step edge

a	a	a	b	b	b	b	b	b	b
a	a	a	a	b	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	a	b	b	b	b
a	a	a	a	a	a	a	b	b	b

Diagonal step edge

a	a	a	a	a	a	a	a	a	a
a	a	a	a	a	a	a	a	a	a
a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b

Corner step edge

a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b

Vertical ramp edge

a	a	c	b	b	b	b	b	b	b
a	a	a	c	b	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	a	c	b	b	b	b
a	a	a	a	a	a	c	b	b	b

Diagonal ramp edge

a	a	a	a	a	a	a	a	a	a
a	a	a	a	c	c	c	c	c	c
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b

Corner ramp edge

Single pixel transition

a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b

Vertical ramp edge

a	a	d	e	b	b	b	b	b	b
a	a	a	d	e	b	b	b	b	b
a	a	a	a	d	e	b	b	b	b
a	a	a	a	a	d	e	b	b	b
a	a	a	a	a	a	d	e	b	b

Diagonal ramp edge

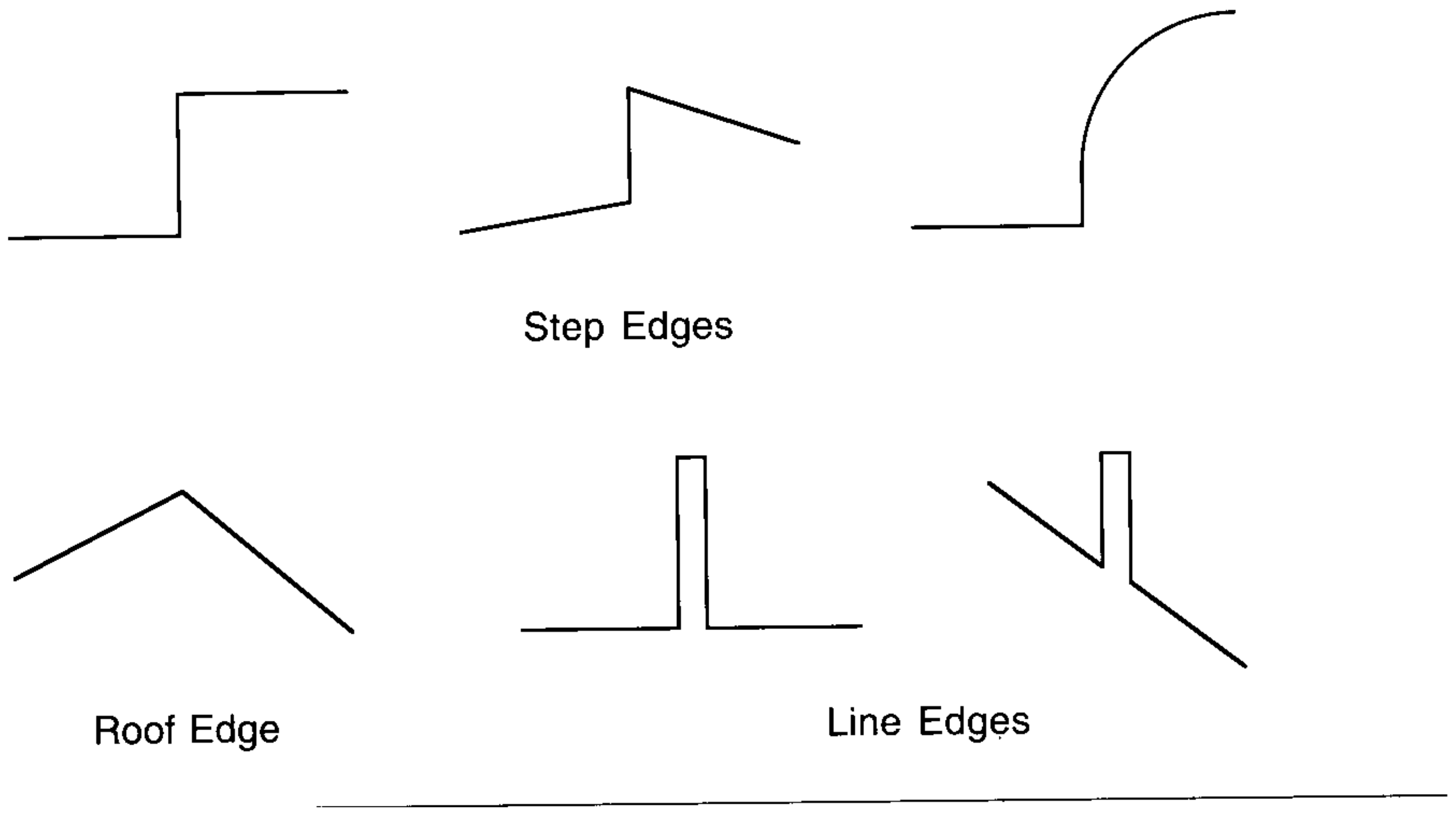
a	a	a	a	a	a	a	a	a	a
a	a	a	a	d	c	c	c	c	c
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b

Corner ramp edge

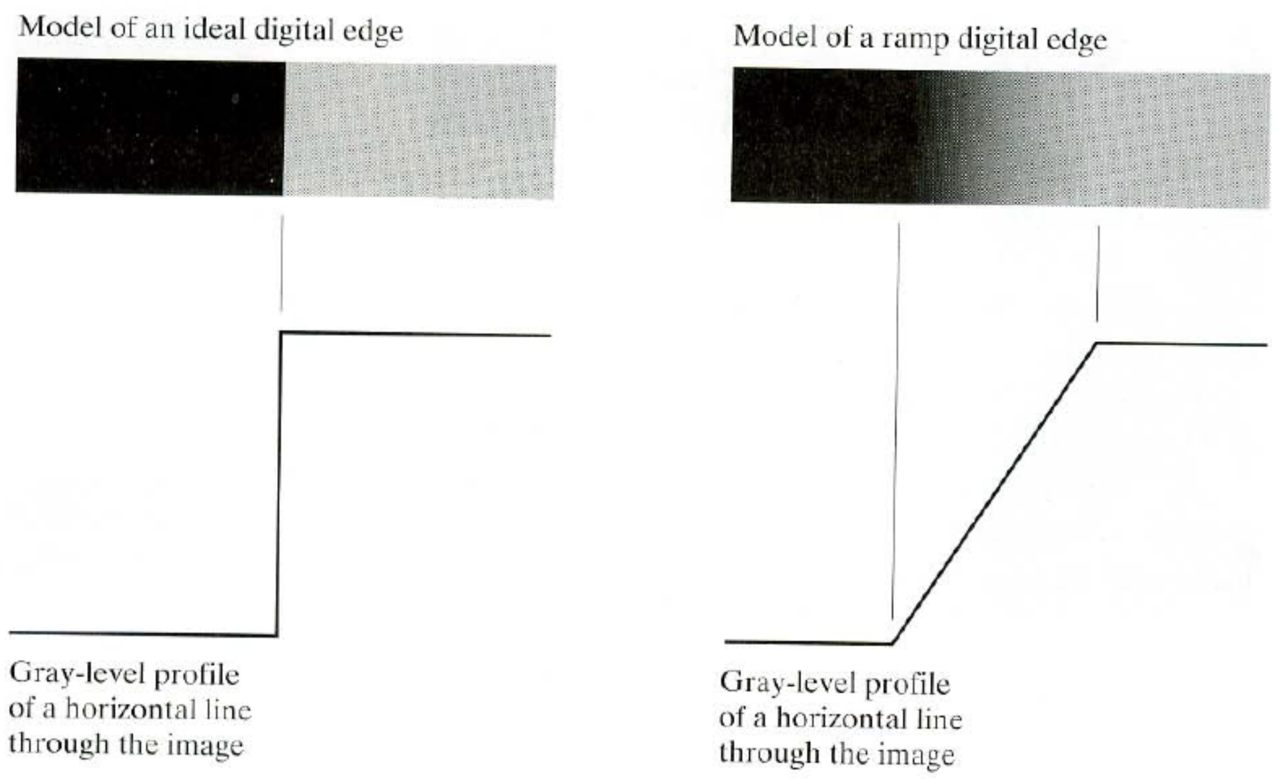
Smoothed transition

$c = \frac{a+b}{2}$ $d = \frac{3a+b}{4}$ $e = \frac{a+3b}{4}$ $b > a$

Profiles of image intensity edges



Models of an ideal digital edge



Model of an ideal digital edge
Gray-level profile of a horizontal line through the image
Model of a ramp digital edge
Gray-level profile of a horizontal line through the image

a b

FIGURE 10.5 (a) Model of an ideal digital edge. (b) Model of a ramp edge. The slope of the ramp is proportional to the degree of blurring in the edge.

Types of edge detectors

- *Unsupervised* or autonomous: only rely on local image features
  - No contextual information is accounted for
  - Simple to implement, flexible, suitable for generic applications
  - Not robust
- *Supervised or contextual*: exploit other sources of information
  - Some a-priori knowledge on the semantics of the scene
  - Output of other kind of processing
  - Less flexible
  - More robust
- There is no golden rule: the choice of the edge detection strategy depends on the application

Types of edge detection

- Differential detection
  - Differential operators are applied to the original image F(x,y) to produce a differential image G(x,y) with accentuated spatial amplitude changes
  - Thresholds are applied to select locations of large amplitude
- Model fitting
  - Fitting a local region of pixel values to a model of the edge, line or spot
- A binary indicator map E(x,y) is used to indicate the positions of edges, lines or points

Differential edge detection

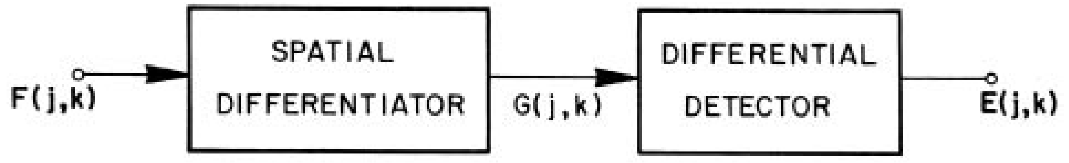
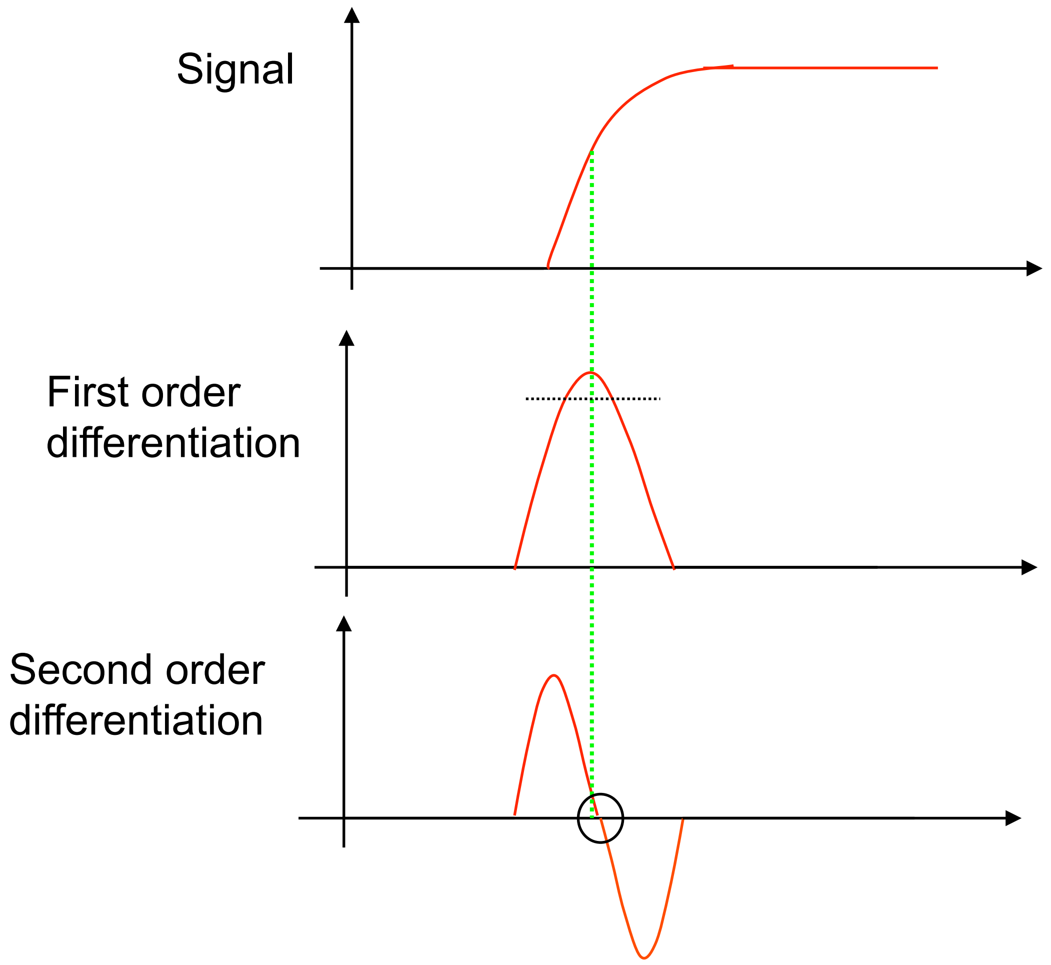


FIGURE 15.1-6. Differential edge, line, and spot detection.

- First order derivatives
- Second order derivatives

Differential edge detection



Gradient thresholding

Zero-crossing

Diff. edge det.: Approach

1. Smoothing of the image
   - To reduce the impact of noise and the number of spurious (non meaningful) edges
   - To regularize the differentiation
2. Calculation of first and second order derivatives
   - Isolation of high spatial frequencies
   - Required features: invariance to rotations, linearity
   - Critical point: choice of the scale (size of the support)
3. Labeling
   - Plausibility measure for the detected point belonging to a contour (to get rid of false edges)

Image gradient

- The *gradient* of an image

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

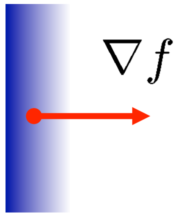
- The gradient *points in the direction of most rapid change in intensity*
- The gradient *direction* is given by

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

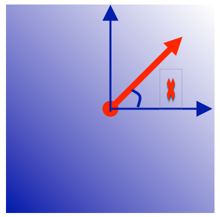
- The *edge strength* is given by the gradient magnitude

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

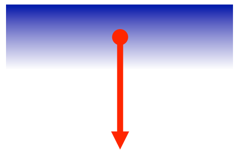
Gradient vector



$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$



$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$



$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$$

Orthogonal gradient vector

- Continuous 1D gradient along a line normal to the edge slope

$$G(x,y) = \frac{\partial f}{\partial x}\cos\theta + \frac{\partial f}{\partial y}\sin\theta$$

- Need of a discrete approximation: definition of a row and a column gradient combined in a spatial gradient amplitude

$$G[j,k] = \left(\left|G_{row}[j,k]\right|^2 + \left|G_{col}[j,k]\right|^2 \right)^{1/2}$$

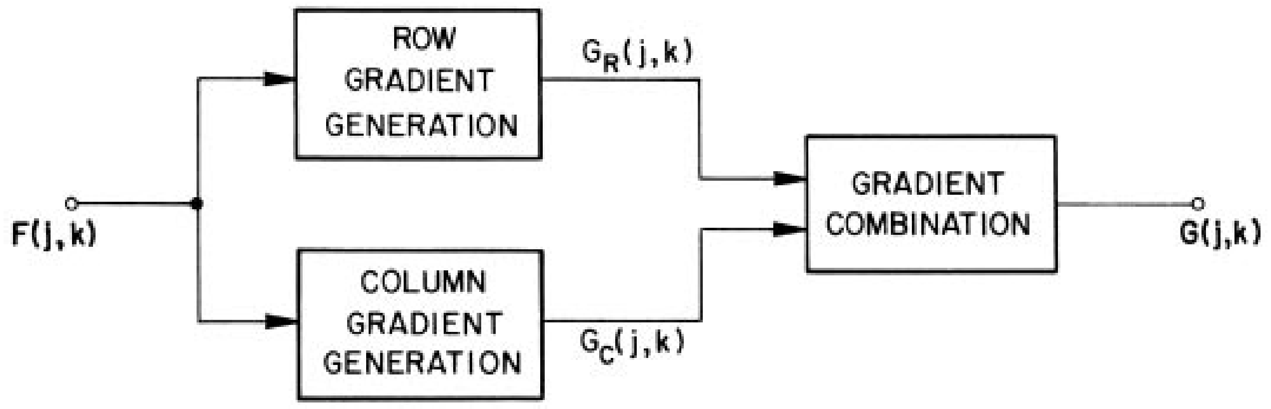
$$G[j,k] = \left|G_{row}[j,k]\right| + \left|G_{col}[j,k]\right|$$

computationally more efficient

$$\vartheta[j,k] = \arctan\left\{ \frac{G_{col}[j,k]}{G_{row}[j,k]} \right\}$$

orientation of the spatial gradient with respect to the row axis

Discrete orthogonal gradient vector

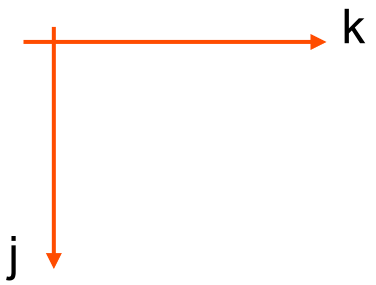


Simplest row/col gradient approximations

$$G_{row}[j,k] \cong f[j,k] - f[j,k-1]$$
$$G_{col}[j,k] \cong f[j,k] - f[j+1,k]$$

$$\begin{array}{cc} & k \\ \boxed{-1} & \boxed{1} \end{array}$$

$$\begin{array}{c} \boxed{1} \\ \boxed{-1} \end{array}$$



$$\begin{array}{lccccccccc} & & & & & k \downarrow & & & & \\ \text{vertical step edge model:} & a & a & a & a & b & b & b & b & b \\ & 0 & 0 & 0 & 0 & h & 0 & 0 & 0 & 0 \end{array}$$

vertical ramp edge model: a a a a c b b b b

0 0 0 0 h/2 h/2 0 0 0

c=(a+b)/2

the edge is not located at the midpoint of the ramp

$$G_{row}[j,k] \cong f[j,k+1] - f[j,k-1]$$
$$G_{col}[j,k] \cong f[j-1,k] - f[j+1,k]$$

0 0 h/2 h h/2 0 0

The discrete gradient

- How can we differentiate a digital image f[x,y]?
  - Option 1: reconstruct a continuous image, then take gradient
  - Option 2: take *discrete derivative* (finite difference)

$$\frac{\partial f[x,y]}{\partial x} = f[x+1,y] - f[x,y]$$

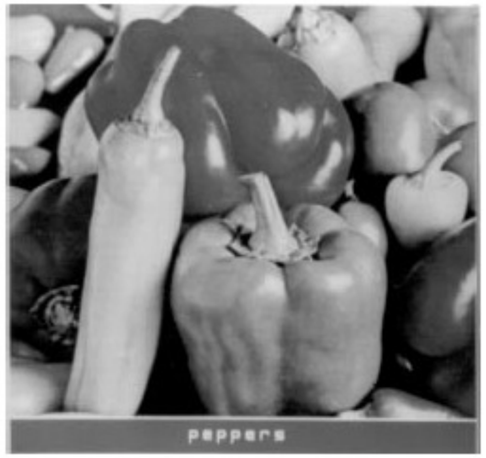
$$\frac{\partial f[x,y]}{\partial y} = f[x,y+1] - f[x,y]$$

  - Discrete approximation

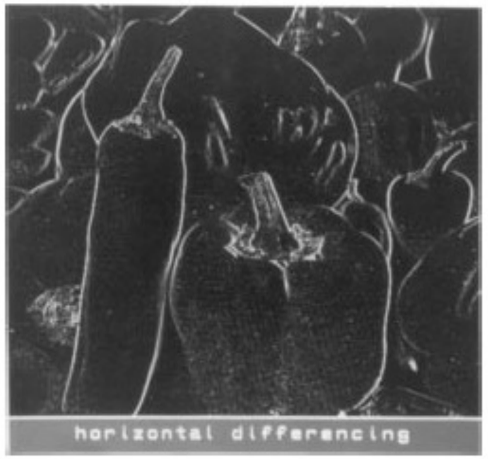
0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

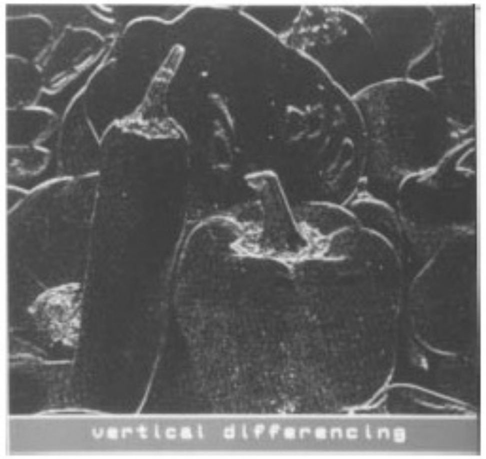
Example



(*a*) Original



(*b*) Horizontal magnitude



(*c*) Vertical magnitude

FIGURE 15.2-2. Horizontal and vertical differencing gradients of the `peppers_mon` image.

Diagonal gradients

- Robert’s cross-difference operator

$$G[j,k] = \left(\left|G_R[j,k]\right|^2 + \left|G_C[j,k]\right|^2\right)^{1/2}$$

square root form

$$G[j,k] = \left|G_R[j,k]\right| + \left|G_C[j,k]\right|$$

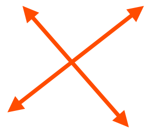
magnitude form

$$\vartheta[j,k] = \frac{\pi}{4} \arctan\left\{\frac{G_C[j,k]}{G_R[j,k]}\right\}$$

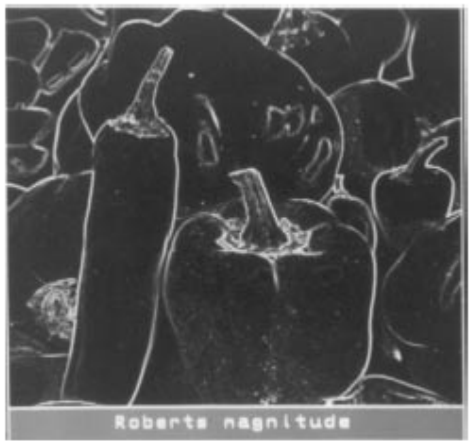
edge orientation with respect to the row axis

$$G_R[j,k] = f[j,k] - f[j+1,k+1]$$

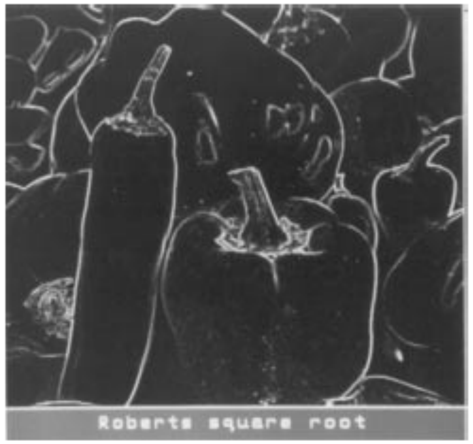
$$G_C[j,k] = f[j,k+1] - f[j+1,k]$$



Example: Robert’s



(*a*) Magnitude



(*b*) Square root

FIGURE 15.2-3. Roberts gradients of the `peppers_mon` image.

Orthogonal differential gradient edge op.

Pixel difference	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Separated pixel difference	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
Roberts	$\begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Prewitt	$\frac{1}{3}\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$
Sobel	$\frac{1}{4}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{4}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$
Frei–Chen	$\frac{1}{2+\sqrt{2}}\begin{bmatrix} 1 & 0 & -1 \\ \sqrt{2} & 0 & -\sqrt{2} \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{2+\sqrt{2}}\begin{bmatrix} -1 & -\sqrt{2} & -1 \\ 0 & 0 & 0 \\ 1 & \sqrt{2} & 1 \end{bmatrix}$

Gradients as convolutions

- The gradient calculation is a neighborhood operation, so it can be put in matrix notations

$$G_{row}[j,k] = f[j,k] * H_{row}[j,k]$$
$$G_{col}[j,k] = f[j,k] * H_{col}[j,k]$$

  – $H_{row/col}$: row and column impulse response arrays

- The size of the convolution kernels can be increased to improve robustness to noise
- Example: normalized boxcar operator

$$\mathbf{H}_R = \frac{1}{21}\begin{bmatrix} 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \end{bmatrix}$$

Gradient filters

- Pixel differences

1	-1

- Symmetric differences

1	0	-1

- Roberts

0	-1
1	0

$H_V = H_H^T$

H_V detects vertical edges

H_H detects horizontal edges

- Prewitt

1/3

1	0	-1
1	0	-1
1	0	-1

H_V: detecting vertical edges

- Sobel

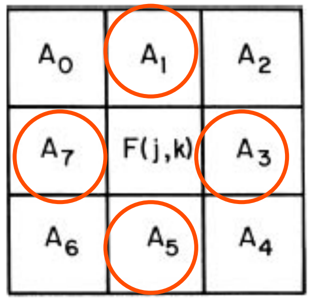
1/4

1	0	-1
2	0	-2
1	0	-1

The filter along the y direction is obtained by transposition of that along the x direction

Introducing averaging

- Differential methods are highly sensitive to small luminance fluctuations → combine with averaging



- Prewitt operator *square root gradient*

k=1

$$G(j,k) = [[G_R(j,k)]^2 + [G_C(j,k)]^2]^{1/2}$$

$$G_R(j,k) = \frac{1}{K+2}[(A_2 + KA_3 + A_4) - (A_0 + KA_7 + A_6)]$$

$$G_C(j,k) = \frac{1}{K+2}[(A_0 + KA_1 + A_2) - (A_6 + KA_5 + A_4)]$$

Sobel, Frei&Chen operator

- Sobel: same as Prewitt with k=2
  - Give the same importance to each pixel in terms to its contribution to spatial gradient

- Frei&Chen: same as Prewitt with k=sqrt(2)
  - The gradient is the same for horizontal, vertical and diagonal edges

Sobel

G_{row} =(1/4)

1	0	-1
2	0	-2
1	0	-1

G_{col} =(1/4)

-1	-2	-1
0	0	0
1	2	1

where

a_0	a_1	a_2
a_7	(i,j)	a_3
a_6	a_5	a_4

Special case of the general one hereafter with c=2

$$G_{row}\,[i,j] = (a0 + c\ a7 + a6) - (a2 + c\ a3 + a4)$$
$$G_{col} = (a6 + c\ a5 + a4) - (a0 + c\ a1 + a2)$$
$$c = 2$$

$$G = \sqrt{G_{row}^2 + G_{col}^2}$$

Sobel extentions

truncated pyramid

$$G_{\text{row}} = k \begin{bmatrix} 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 2 & 2 & 0 & -2 & -2 & -1 \\ 1 & 2 & 3 & 0 & -3 & -2 & -1 \\ 1 & 2 & 3 & 0 & -3 & -2 & -1 \\ 1 & 2 & 3 & 0 & -3 & -2 & -1 \\ 1 & 2 & 2 & 0 & -2 & -2 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \end{bmatrix}$$

Sobel 7x7

$$G_{\text{col}} = \left[\begin{array}{cc|ccc|cc} -1 & -1 & -1 & -2 & -1 & -1 & -1 \\ -1 & -1 & -1 & -2 & -1 & -1 & -1 \\ \cline{3-5} -1 & -1 & -1 & -2 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 2 & 1 & 1 & 1 \\ \cline{3-5} 1 & 1 & 1 & 2 & 1 & 1 & 1 \\ 1 & 1 & 1 & 2 & 1 & 1 & 1 \end{array}\right]$$

Prewitt

$$G_{row} = 1/3 \begin{array}{|c|c|c|} \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \qquad G_{col} = 1/3 \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline \end{array} \qquad c = 1$$

- Kirsch operator
  - 8 directional masks, each selecting one specific direction
  - “winner takes all” paradigm for the absolute value of the gradient and direction selected by the index of the corresponding mask
- Robinson operator
  - 8 directional masks, similar to Kirsh

Directional masks

$$S_1 = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline\end{array}$$

$$S_2 = \begin{array}{|c|c|c|}\hline 0 & 1 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & -1 & 0 \\ \hline\end{array}$$

$$S_3 = \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline\end{array}$$

$$S_4 = \begin{array}{|c|c|c|}\hline 1 & 1 & 0 \\ \hline 1 & 0 & -1 \\ \hline 0 & -1 & -1 \\ \hline\end{array}$$

$$S_5 = \begin{array}{|c|c|c|}\hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline\end{array}$$

$$S_6 = \begin{array}{|c|c|c|}\hline 0 & -1 & -1 \\ \hline 1 & 0 & -1 \\ \hline 1 & 1 & 0 \\ \hline\end{array}$$

$$S_7 = \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 1 & 1 \\ \hline\end{array}$$

$$S_8 = \begin{array}{|c|c|c|}\hline -1 & -1 & 0 \\ \hline -1 & 0 & 1 \\ \hline 0 & 1 & 1 \\ \hline\end{array}$$

Kirsh operator

- The Kirsch operator or Kirsch compass kernel is a non-linear edge detector that finds the maximum edge strength in a few predetermined directions.
- The operator takes a single kernel mask and rotates it in 45 degree increments through all 8 compass directions: N, NW, W, SW, S, SE, E, and NE. The edge magnitude of the Kirsch operator is calculated as the maximum magnitude across all directions:

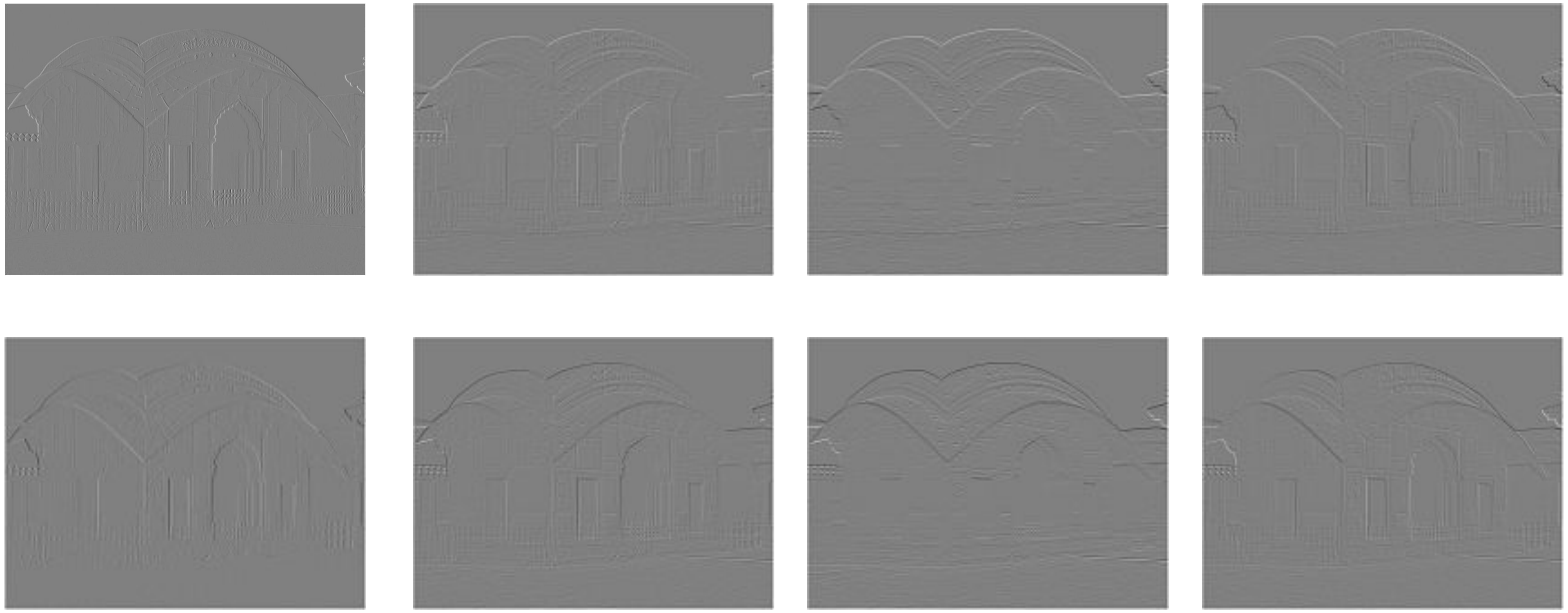
$$imf[n,m] = \max_z \left\{ imf^z[m,n] \right\}$$

$$\Downarrow$$

Edge image in direction z

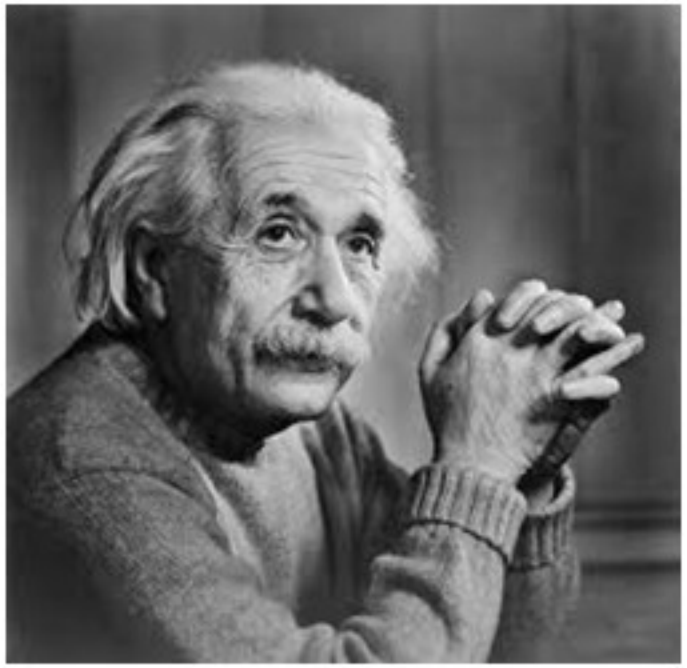
Kirsh operator

$$h^1 = \begin{bmatrix} +5 & +5 & +5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \quad h^2 = \begin{bmatrix} +5 & +5 & -3 \\ +5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \quad h^3 = \begin{bmatrix} +5 & -3 & -3 \\ +5 & 0 & -3 \\ +5 & -3 & -3 \end{bmatrix} \quad h^4 = \begin{bmatrix} -3 & -3 & -3 \\ +5 & 0 & -3 \\ +5 & +5 & -3 \end{bmatrix}$$

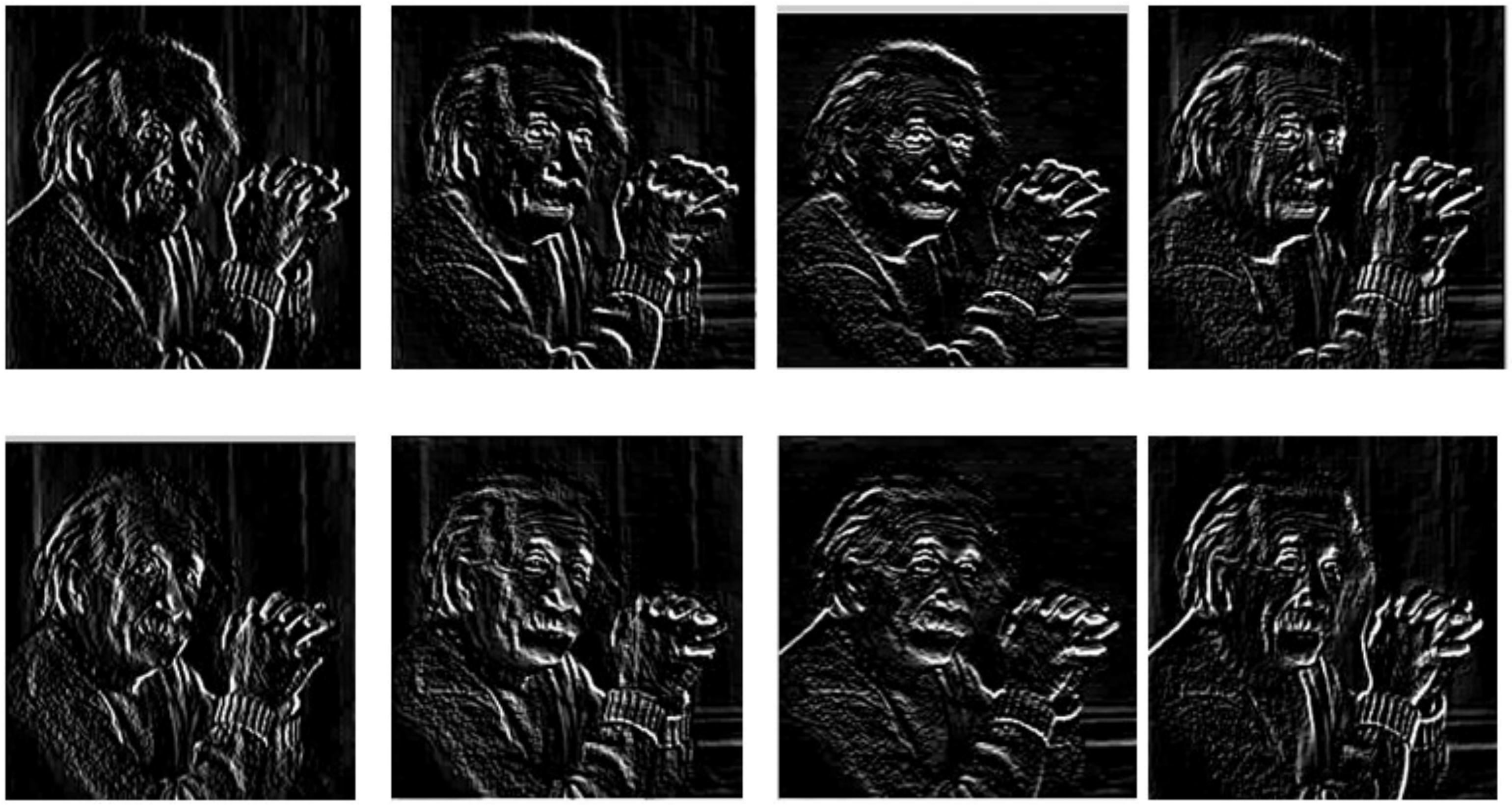


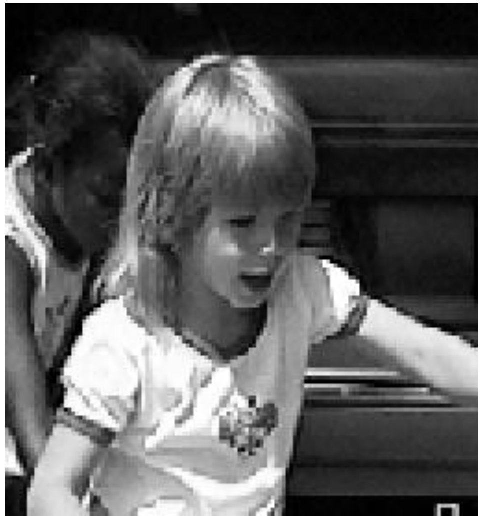
Robinson operator

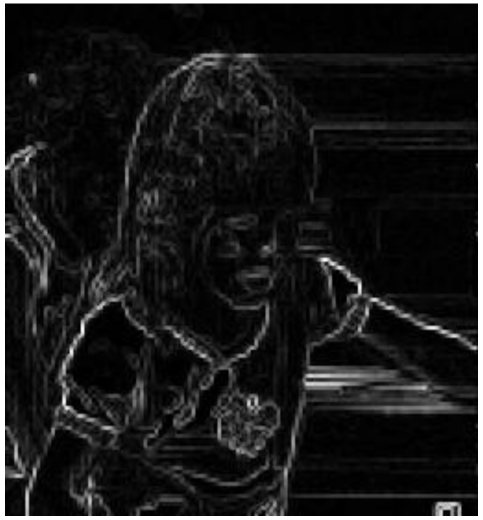
$$h^1 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad h^2 = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} \quad h^3 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad h^4 = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{bmatrix}$$



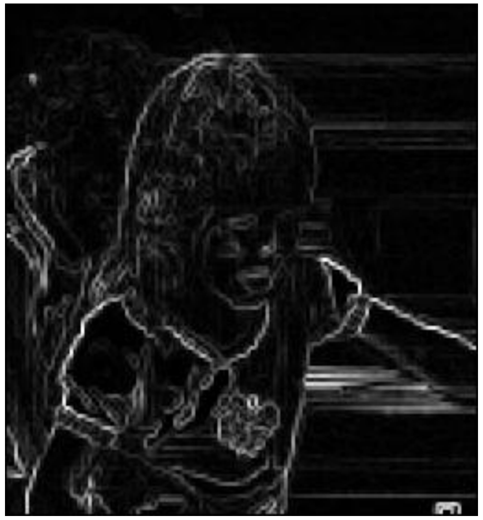
Robinson operator



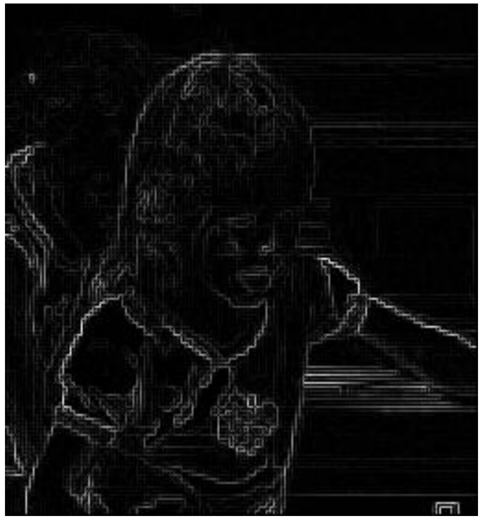




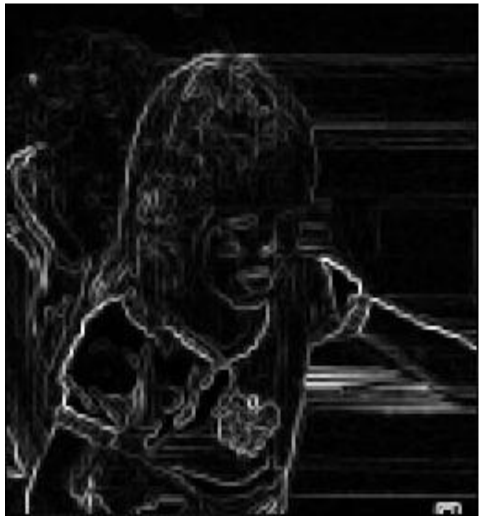
Sobel



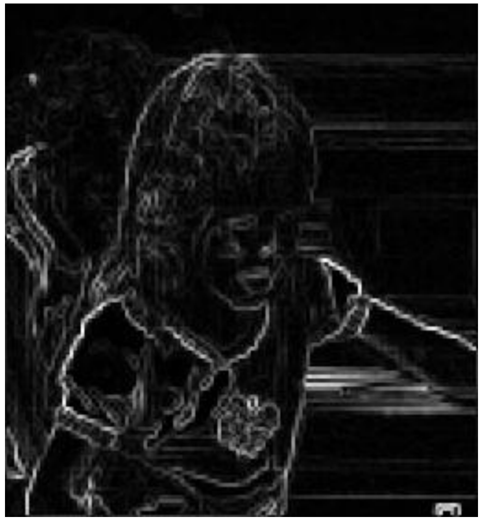
Prewitt



Roberts

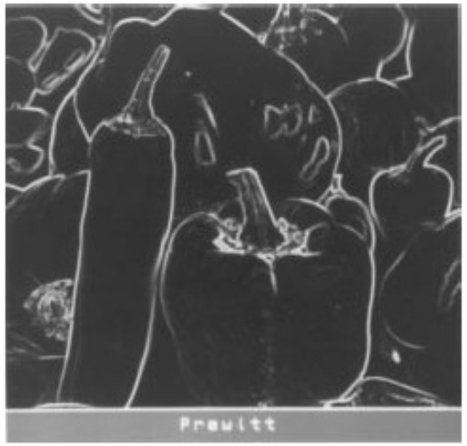


Kirsch

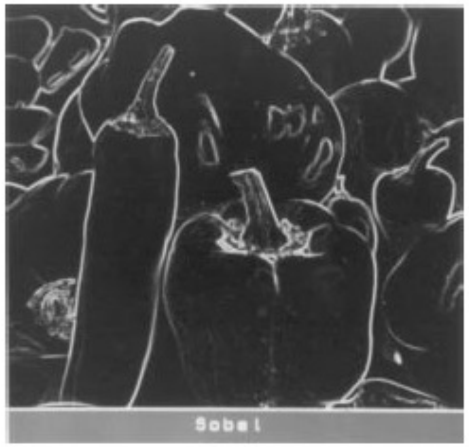


Robinson

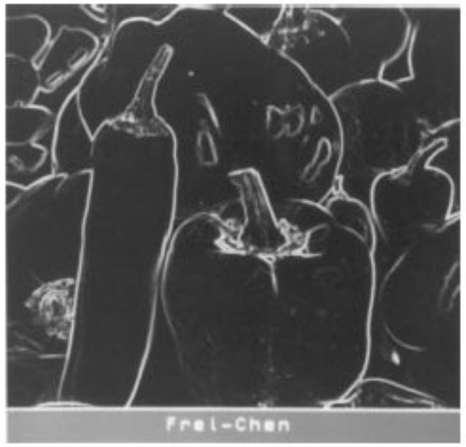
Comparison



(*a*) Prewitt



(*b*) Sobel



(*c*) Frei–Chen

Truncated pyramid op.

- Linearly decreasing weighting to pixels away from the center of the edge

$$\mathbf{H}_R = \frac{1}{34}\begin{bmatrix} 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 2 & 2 & 0 & -2 & -2 & -1 \\ 1 & 2 & 3 & 0 & -3 & -2 & -1 \\ 1 & 2 & 3 & 0 & -3 & -2 & -1 \\ 1 & 2 & 3 & 0 & -3 & -2 & -1 \\ 1 & 2 & 2 & 0 & -2 & -2 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \end{bmatrix}$$

Edge detection in presence of noise

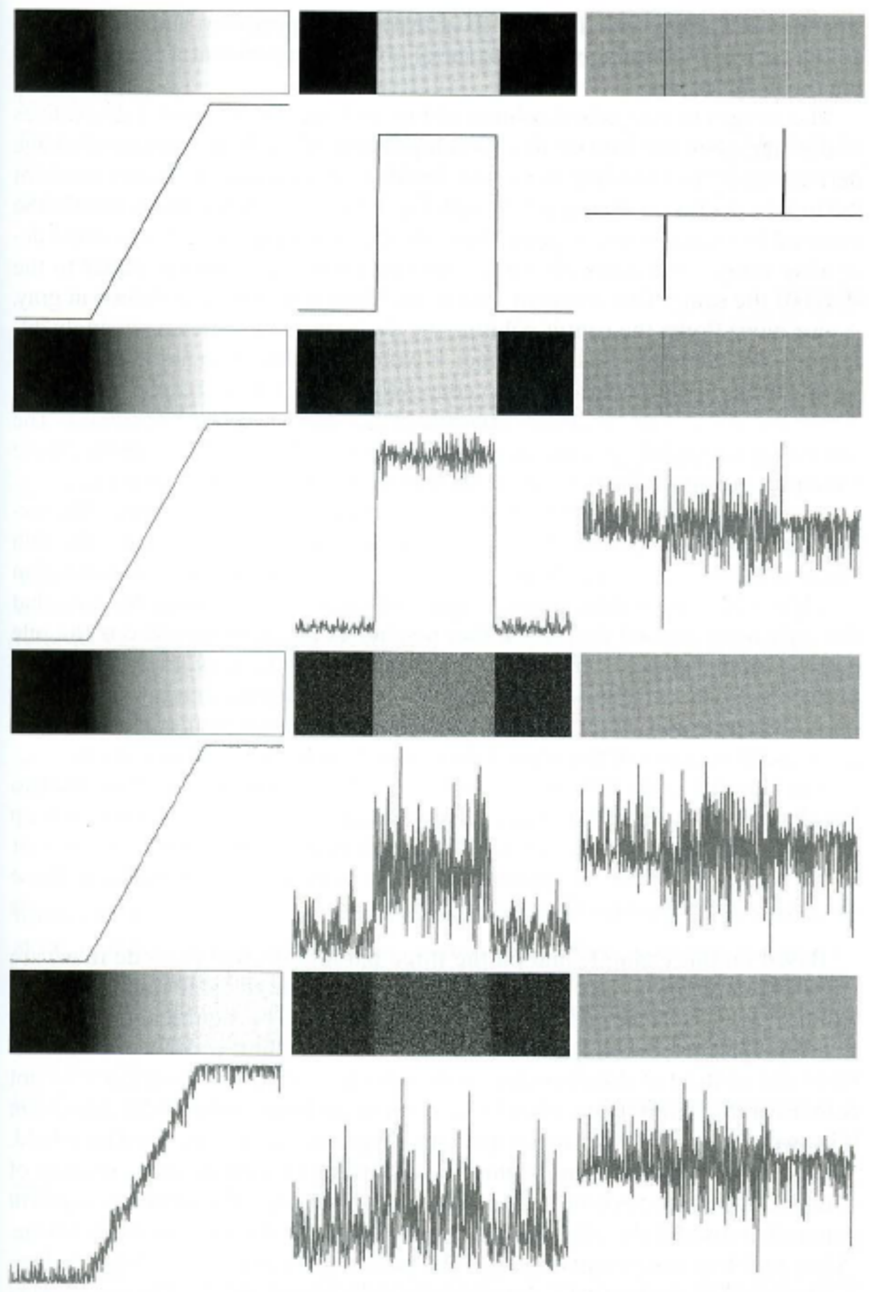


FIGURE 10.7 First column: images and gray-level profiles of a ramp edge corrupted by random Gaussian noise of mean 0 and σ = 0.0, 0.1, 1.0, and 10.0, respectively. Second column: first-derivative images and gray-level profiles. Third column: second-derivative images and gray-level profiles.

a
b
c
d

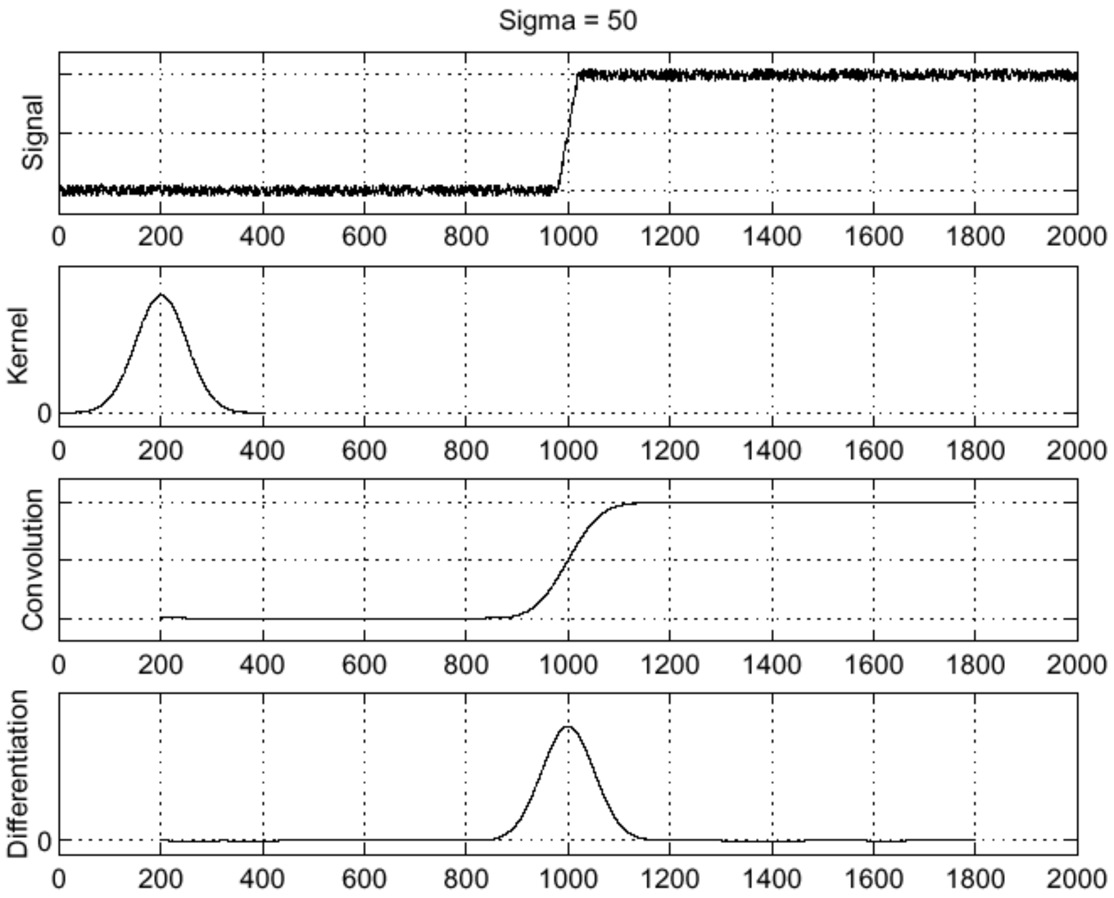
Improving robustness to noise

- Combining smoothing with differentiation
  - Solution 1: do smoothing first and then differentiation
  - Solution 2: differentiate the smoothing filter and do filtering

$$\boxed{\frac{d}{dx}\left(h * f\right) = \frac{dh}{dx} * f} = h * \frac{df}{dx}$$

h: smoothing filter

Solution 1: Smoothing+Differentiation



h

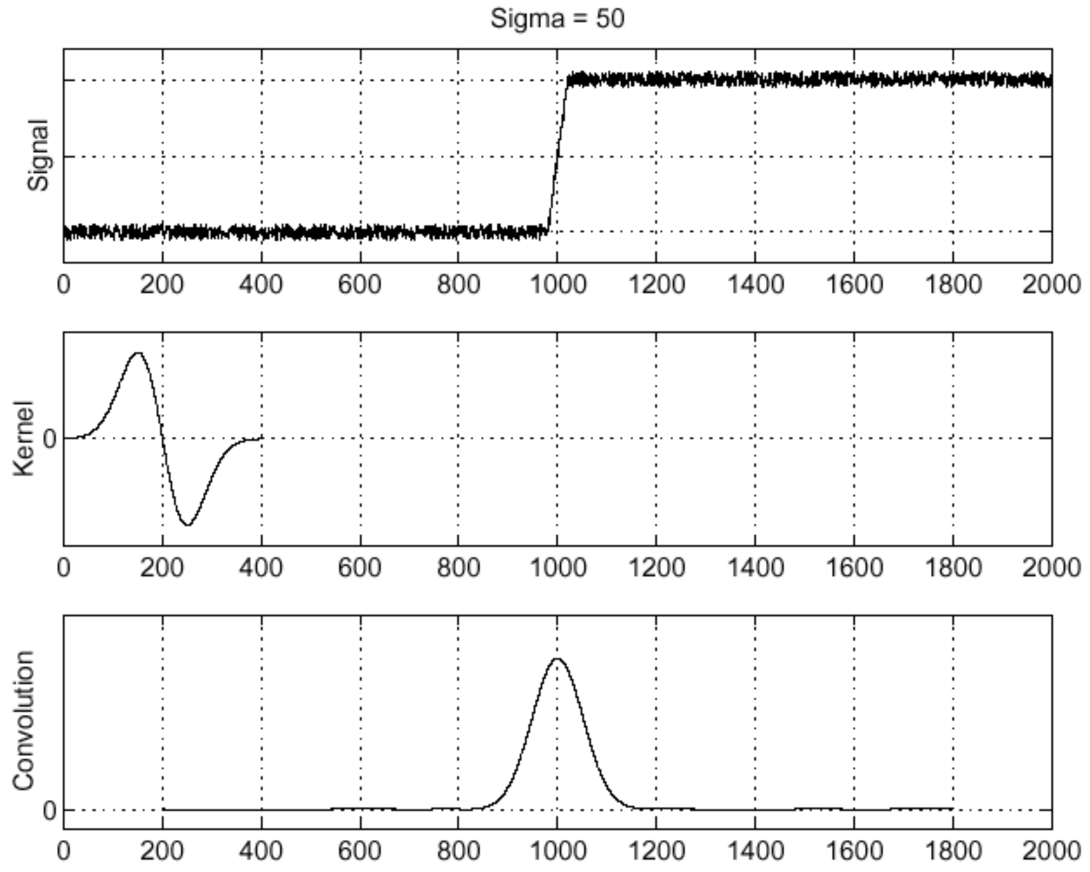
h*f

$$\frac{\partial}{\partial x}(h \star f)$$

Look for peaks in $\frac{\partial}{\partial x}(h \star f)$

Sol. 2: Differentiation of the smoothing filter

$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$



$$\frac{\partial}{\partial x}h$$

$$(\frac{\partial}{\partial x}h) \star f$$

Extending to 2° order derivative

- The derivative of a convolution is equal to the convolution of either of the functions with the derivative of the other

$$z(x) = f(x) * g(x)$$
$$\frac{dz}{dx} = \frac{df}{dx} * g = f * \frac{dg}{dx}$$

- Iterating

$$z(x) = f(x) * g(x)$$
$$\frac{d^2 z}{dx^2} = \frac{d}{dx}\left(\frac{df}{dx} * g\right) = \frac{d^2 f}{dx^2} * g$$

Hints of the proof

- Intuition (OP)

$$c(t) = f(t) * g(t) = f * g(t) = \int_{-\infty}^{+\infty} f(\tau) g(t-\tau) d\tau$$

$$c'(t) = \frac{dc(t)}{dt} = \frac{d}{dt}\left(f(t) * g(t)\right)$$

$$C(\omega) = \Im\{c(t)\} = \Im\{f(t) * g(t)\} = F(\omega) G(\omega)$$

$$\Im\{c'(t)\} = j\omega \Im\{c(t)\} = j\omega F(\omega) G(\omega) = \begin{cases} \left[j\omega F(\omega)\right] G(\omega) \rightarrow f'(t) * g(t) \\ F(\omega)\left[j\omega G(\omega)\right] \rightarrow f(t) * g'(t) \end{cases}$$

Remark

- The order in which differentiation and smoothing are performed depends on their properties.
  - Such operations are interchangeable as long as they are linear. Thus, if both smoothing and differentiation are performed by linear operators they are interchangeable
  - In this case they can be performed at the same time by filtering the image with the differentiation of the smoothing filter

- Argyle and Macleod
- Laplacian of Gaussian (LoG)
  - Difference of Gaussians (DoG)

Argyle and Macleod

- Use a large neighborhood Gaussian-shaped weighting for noise suppression

$$g(x, s) = [2\pi s^2]^{-1/2} \exp\{-1/2(x/s)^2\}$$

Argyle operator horizontal coordinate impulse response array can be expressed as a sampled version of the continuous domain impulse response. s and t are the spread parameters

smoothing+differentiation

Argyle operator horizontal coordinate impulse response array

Argyle

$$H_R(j, k) = \begin{cases} -2g(x, s)g(y, t) & \text{for } x \geq 0 \\ 2g(x, s)g(y, t) & \text{for } x < 0 \end{cases}$$

smoothing along the edge and differentiation along the orthogonal direction

McLeod

$$H_R(j, k) = [g(x+s, s) - g(x-s, s)]g(y, t)$$

The Argyle and Macleod operators, unlike the boxcar operator, give decreasing importance to pixels far removed from the center of the neighborhood.

Argyle and Macleod

- Extended-size differential gradient operators can be considered to be compound operators in which a smoothing operation is performed on a noisy image followed by a differentiation operation.
- The compound gradient impulse response can be written as

$$H(j,k) = \underbrace{H_G(j,k)}_{\text{gradient op.}} \circledast \underbrace{H_S(j,k)}_{\text{low pass}}$$

- Example

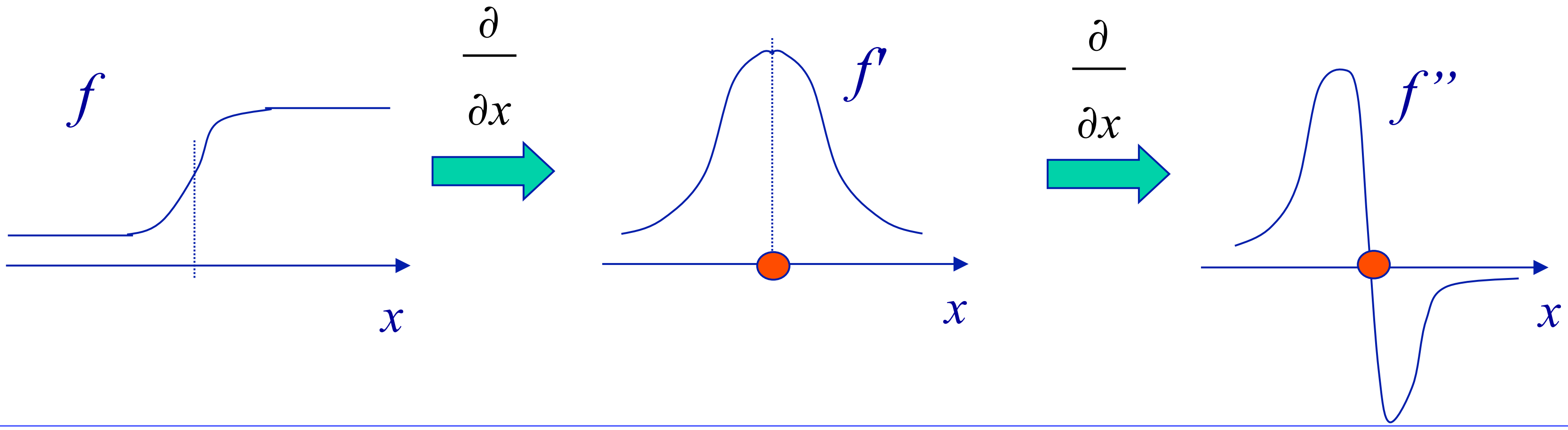
if Hg is the 3x3 Prewitt row gradient operator and Hs (j,k) =1/9, for all (j,k) in a 3x3 matrix, is a uniform smoothing operator, the resultant row gradient operator, after normalization to unit positive and negative gain, becomes

[1 1 1
0 0 0
-1 -1 -1]

$$\mathbf{H}_R = \frac{1}{18}\begin{bmatrix} 1 & 1 & 0 & -1 & -1 \\ 2 & 2 & 0 & -2 & -2 \\ 3 & 3 & 0 & -3 & -3 \\ 2 & 2 & 0 & -2 & -2 \\ 1 & 1 & 0 & -1 & -1 \end{bmatrix}$$

Second order derivative

- Edge detectors based on first order derivative are not robust
  - High sensitivity to noise, need a threshold
- Second order derivative operators detect the edge at the zero-crossing of the second derivative → more robust, more precise
  - Less sensitive to noise, usually don't need a threshold for post-processing of the contours image



Laplace operator

- Second order differentiation operator

$$\Delta f = \nabla^2 f = \nabla \cdot \nabla f$$

$$\nabla^2 f = \sum_{i=1}^{N} \frac{\partial^2 f}{\partial x_i^2}$$

$$N = 2 \rightarrow \nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

- Directional derivative

$$D_{\vec{v}} f(\vec{x}) = \sum_{i=1}^{N} v_i \frac{\partial f}{\partial x_i}$$

$$v_i = \left\langle \vec{v}, \vec{i} \right\rangle$$

Laplace operator

- Second order derivative in the continuous domain

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

- Discrete approximation

$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial G_x}{\partial x} = \frac{\partial}{\partial x}\left[f(i, j+1) - f(i, j)\right] =$$

$$= \frac{\partial f(i, j+1)}{\partial x} - \frac{\partial f(i, j)}{\partial x} =$$

$$= \left[f(i, j+2) - f(i, j+1)\right] - \left[f(i, j+1) - f(i, j)\right] =$$

$$= f(i, j+2) - 2f(i, j+1) + f(i, j)$$

Discrete approximation: proof

- Centring the estimation on *(i,j)* the simplest approximation is to compute the difference of slopes along each axis

$$G(x,y) = -\nabla^2 f(x,y)$$

$$G_{row}[i,j] = \left(f[i,j] - f[i,j-1]\right) - \left(f[i,j+1] - f[i,j]\right) = 2f[i,j] - f[i,j-1] - f[i,j+1]$$

$$G_{col}[i,j] = \left(f[i,j] - f[i+1,j]\right) - \left(f[i-1,j] - f[i,j]\right) = 2f[i,j] - f[i+1,j] - f[i-1,j]$$

- This can be put in operator and matrix form as

$$G[i,j] = f[i,j] * H[i,j]$$

$$H = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Discrete approximation

- The 4-neighbors Laplacian is often normalized to provide unit gain averages of the positive and negative weighted pixels in the 3x3 neighborhood
- Gain normalized 4-neighbors Laplacian

$$H = \frac{1}{4}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

- The weights of the pixels in the neighborhood, and thus the normalization coefficient, can be changed to emphasize the edges.
  - Ex. Prewitt modified Laplacian

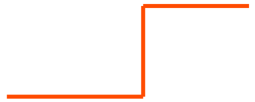
$$H = \frac{1}{8}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Discrete approximation

– Gain normalized separable 8 neighbors Laplacian

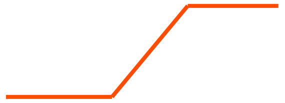
$$H = \frac{1}{8}\begin{bmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{bmatrix}$$

a	a	a	b	b
a	a	a	b	b
a	a	a	b	b
a	a	a	b	b
a	a	a	b	b



$$\begin{array}{ccccccc} a & a & a & a & b & b & b \\ 0 & 0 & 0 & -\frac{3}{8}h & \frac{3}{8}h & 0 & 0 \end{array}$$

h=a-b



$$\begin{array}{ccccccc} a & a & a & c & b & b & b \\ 0 & 0 & -\frac{3}{16}h & 0 & \frac{3}{16}h & 0 & 0 \end{array}$$

Note

- Without sign change after the evaluation of the Laplacian
  - However, the sign is meaningless if we evaluate the modulus of the gradient

$$\frac{\partial^2 f}{\partial x^2} = f(i, j+1) - 2f(i, j) + f(i, j-1)$$

$$\frac{\partial^2 f}{\partial y^2} = f(i+1, j) - 2f(i, j) + f(i-1, j)$$

- Different possible Laplacian matrices

$$\nabla^2 = \begin{array}{|c|c|c|}\hline 0 & 1 & 0 \\ \hline 1 & -4 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array} \qquad \begin{array}{|c|c|c|}\hline 1 & 4 & 1 \\ \hline 4 & -20 & 4 \\ \hline 1 & 4 & 1 \\ \hline \end{array}$$

$$\nabla^2 = \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array} \qquad \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline -1 & 8 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$$

Laplacian of Gaussian

- Quite often the zero crossing does not happen at a pixel location
  - See the example of the step edge
- It is common choice to locate the edge at a pixel with a positive response having a neighbor with a negative response
- Laplacian of Gaussian: Marr&Hildreth have proposed an operator in which Gaussian shaped smoothing is performed *prior* to the application of the Laplacian

➢ Continuous LoG gradient

$$LOG(x,y) = -\nabla^2 \left\{ f(x,y) * H_S(x,y) \right\}$$

$$H_S(x,y) = g(x,s)g(y,s)$$

$$g(x,s) = \frac{1}{\sqrt{2\pi s^2}} \exp\left\{ -\frac{1}{2}\left(\frac{x}{s}\right)^2 \right\}$$

impulse response of the Gaussian smoothing kernel

LoG operator

- As a result of the linearity of the second derivative operator and of the convolution

$$LOG[j,k] = f[j,k] * H[j,k] \qquad (1)$$

$$H(x,y) = -\nabla^2 \left\{ g(x,s)g(y,s) \right\}$$

$$H(x,y) = \frac{1}{\pi s^4}\left(1 - \frac{x^2+y^2}{2s^2}\right)\exp\left\{-\frac{x^2+y^2}{2s^2}\right\}$$

- It can be shown that
  - The convolution (1) can be performed separately along rows and cols
  - It is possible to approximate the LOG impulse response closely by a difference of Gaussians (DOG) operator

$$H(x,y) = g(x,s_1)g(y,s_1) - g(x,s_2)g(y,s_2), \qquad s_1 < s_2$$

The LoG operator

$$g(x,y) = \frac{1}{2\pi s^2} \exp\left[-\frac{x^2+y^2}{2s^2}\right]$$

$$h(x,y) : \nabla^2\left[g(x,y) * f(x,y)\right] = \left[\nabla^2 g(x,y)\right] * f(x,y) = h(x,y) * f(x,y)$$

where

$$\nabla^2 g(x,y) = \frac{x^2+y^2-2s^2}{2\pi s^4} \exp\left[-\frac{x^2+y^2}{2s^2}\right] \quad \text{mexican hat}$$

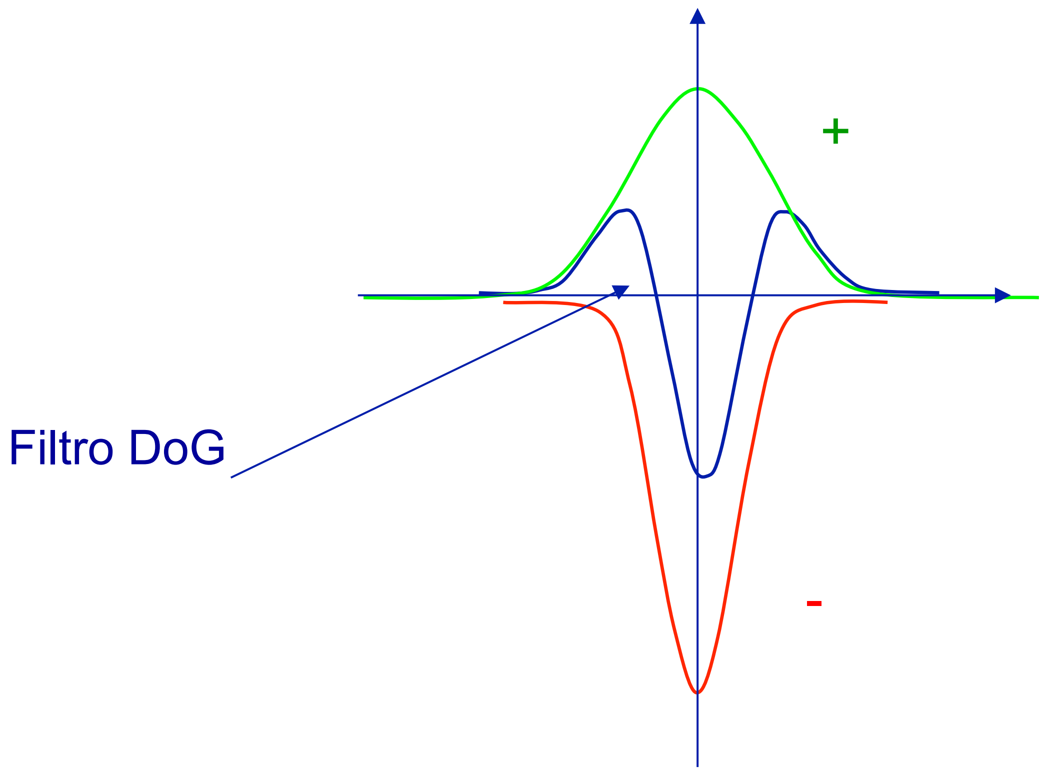
- How to choose s?
  - Large values: pronounced smoothing → better denoising BUT smears out sharp boundaries reducing the precision in edge localization
  - Small values: soft smoothing → lower noise reduction BUT better boundary preservation
  - A good solution could be to follow a multiscale approach (s is the scale)

LoG filtering

- Gaussian smoothing (low-pass filter)
  - Noise reduction (the larger the filter, the higher the smoothing)
  - BUT
    - Smears out edges
    - Blurs the image (defocusing)
- Laplacian detection (high-pass filter)
- Edge location by interpolation
  - The zero-crossing does not happen in a pixel site

LoG filtering = Gaussian smoothing + Laplacian detection

DoG



FDoG

- First derivative of Gaussian op. [Pratt]
  - Gaussian shaped smoothing is followed by differentiation
    - FDoG continuous domain horizontal impulse response

$$H_R(j,k) = \frac{-\partial[g(x,s)g(y,t)]}{\partial x}$$

$$g(x,s) = [2\pi s^2]^{-1/2} \exp\{-1/2(x/s)^2\}$$

$$H_R(j,k) = \frac{-xg(x,s)g(y,t)}{s^2}$$

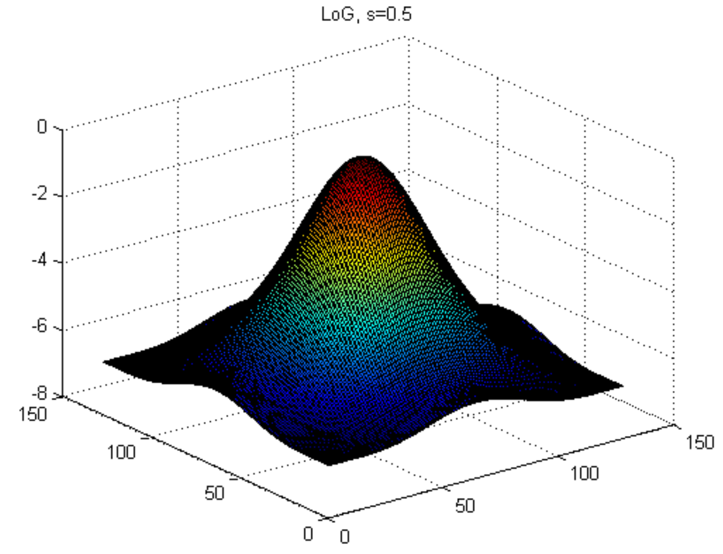
introduces the sign change

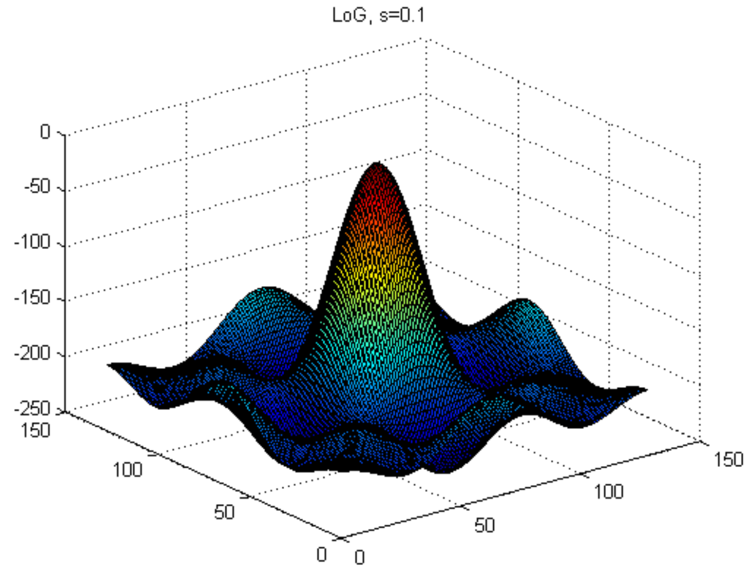
$$H_R(x,y) = -\frac{x}{s^2} g(x,s) g(y,t)$$

$$H_C(x,y) = -\frac{y}{t^2} g(x,s) g(y,t)$$

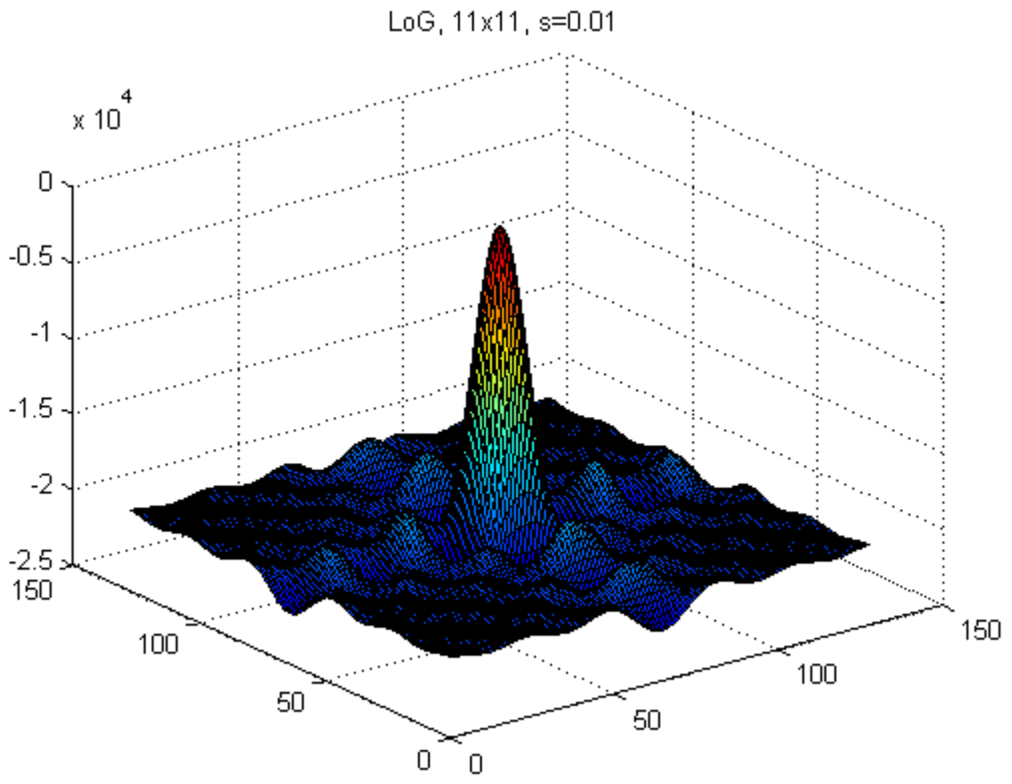
5x5 LoG

$$H[j,k] = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$





11x11 LoG

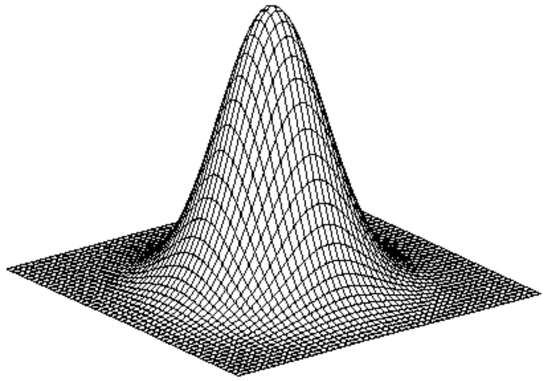


LoG

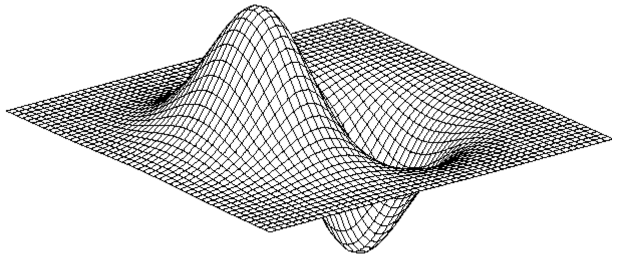
- Independent variables
  - s value: larger values allow larger denoising but smear out details and made contour extraction not quite precise
- Solutions
  - Trade off
  - Multiscale

2D edge detection filters

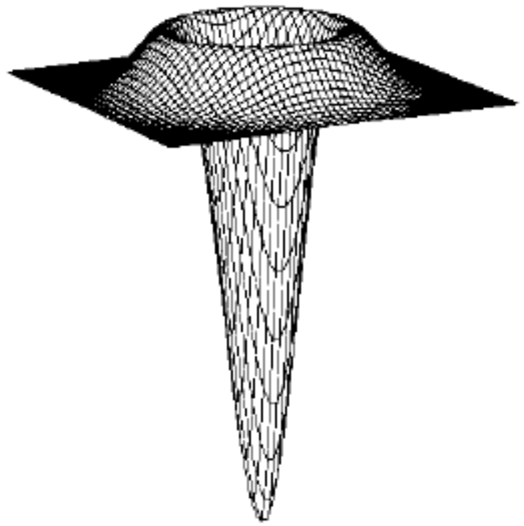
Gaussian



derivative of Gaussian



Laplacian of Gaussian



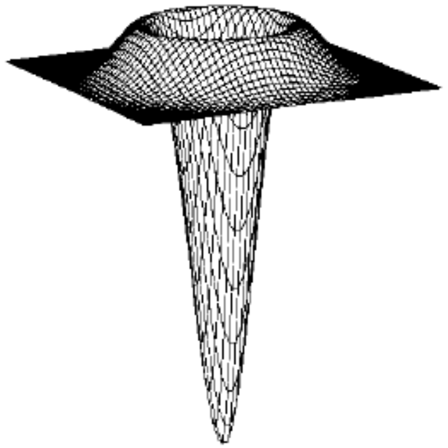
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$

$$\frac{\partial}{\partial u} h_\sigma(u,v)$$

$$\nabla^2 h_\sigma(u,v)$$

LoG: example

- The Laplacian of a Gaussian filter



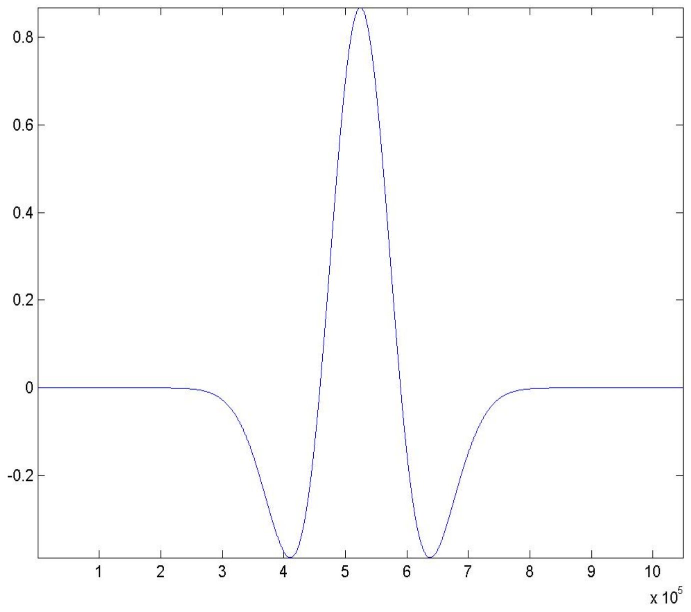
A digital approximation:

0	0	1	0	0
0	1	2	1	0
1	2	-16	2	1
0	1	2	1	0
0	0	1	0	0

Second derivative

- Laplacian of Gaussian: (LoG) – Mexican Hat

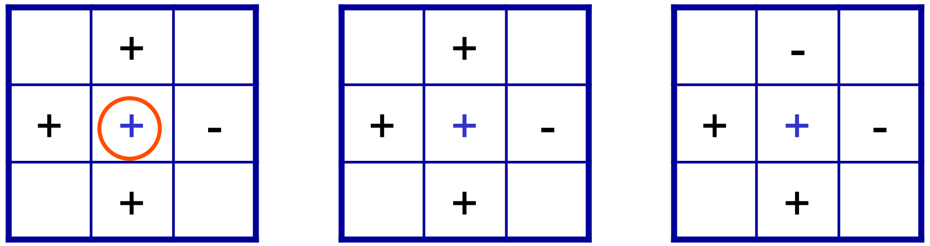
0	1	0
1	-4	1
0	1	0



- Laplacian of Gaussian: Link to early vision: the 2D Mexican Hat closely resembles the receptive fields of simple cells in the retina → *edge detection is one of the first steps in vision*

Laplacian zero-crossing detection

- Zero-valued Laplacian response pixels are unlikely in real images
- Practical solution: form the maximum of all positive Laplacian responses and the minimum of all Laplacian responses in a 3x3 window. If the *difference* between the two exceeds a threshold an edge is assumed to be present.
- Laplacian zero-crossing patterns

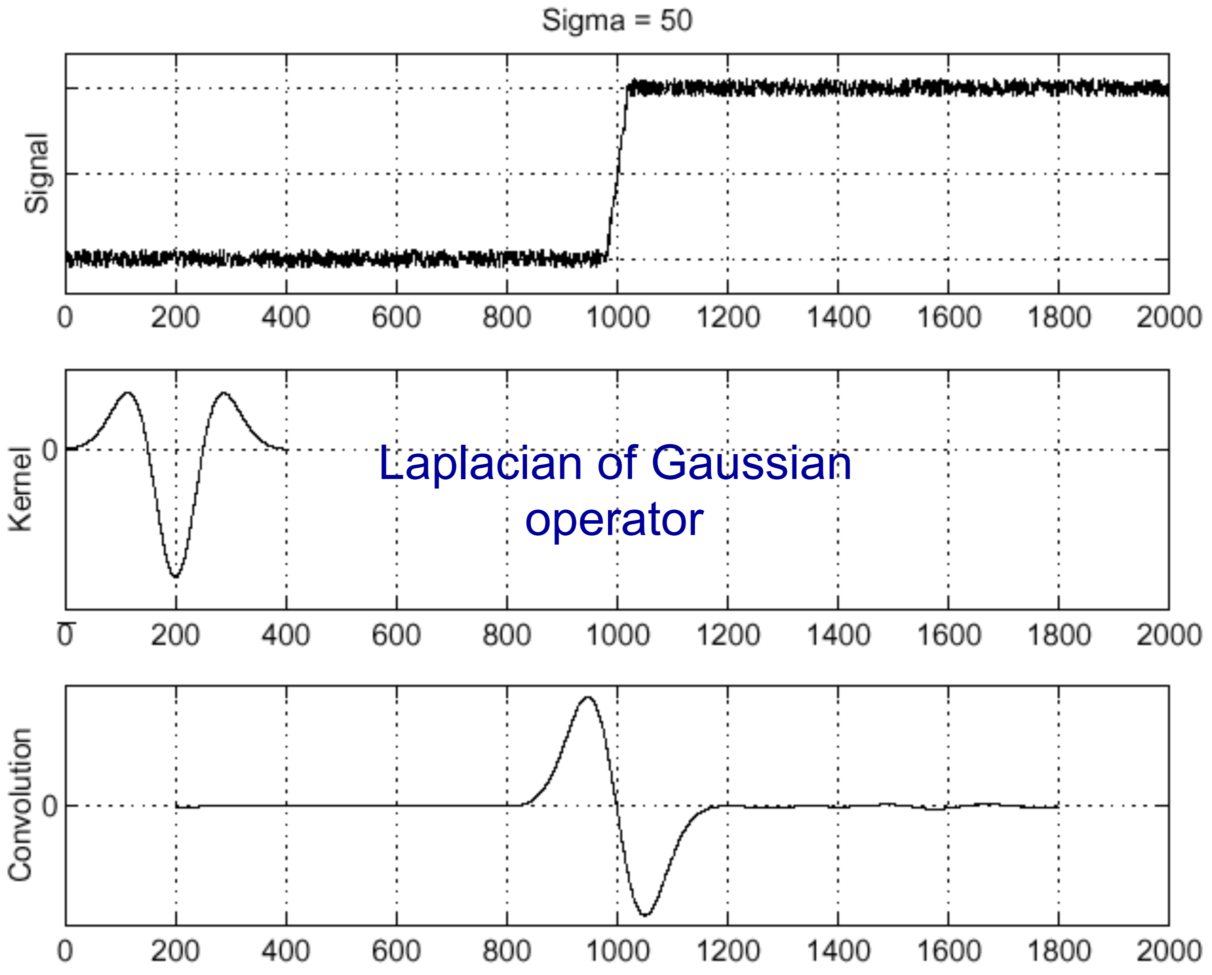


+: zero or positive

Laplacian of Gaussian (LoG)

$$\frac{\partial^2}{\partial x^2}h$$

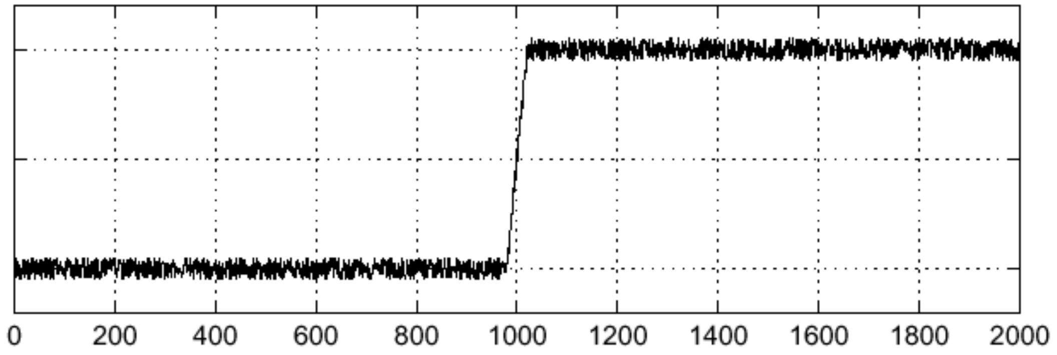
$$\left(\frac{\partial^2}{\partial x^2}h\right) \star f$$

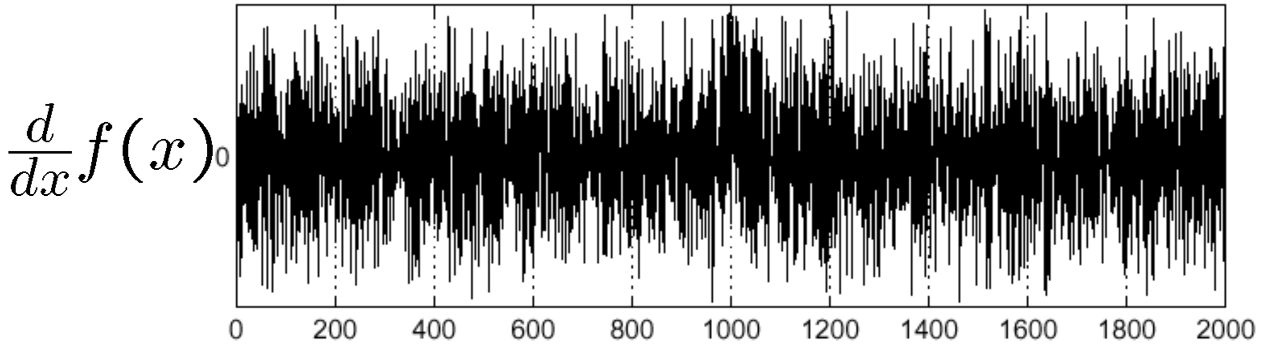


Zero-crossings of bottom graph

Effects of noise

- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a signal

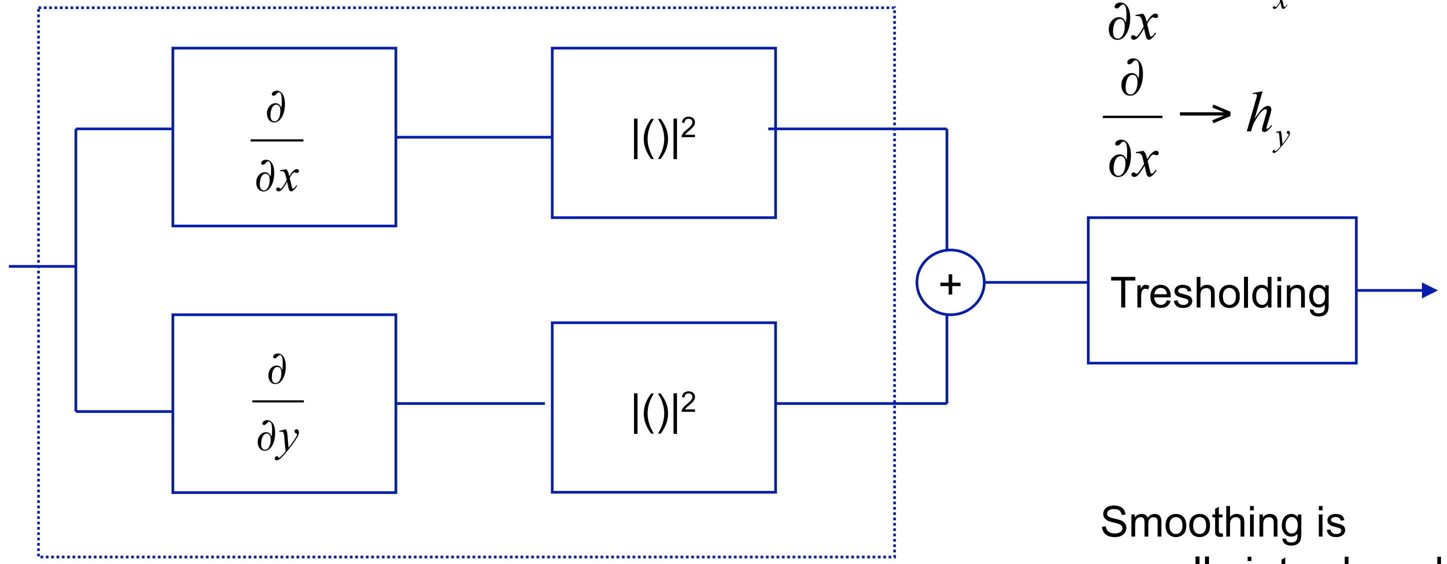




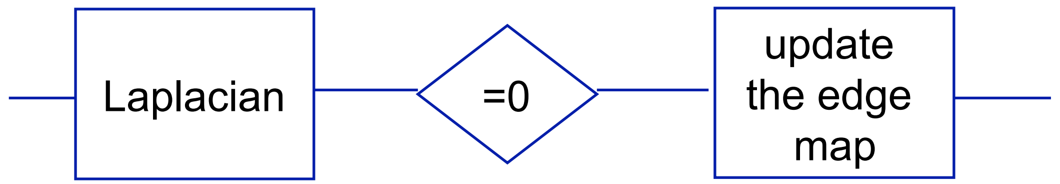
Gradient thresholding

Modulus of the gradient thresholding

$$\frac{\partial}{\partial x} \to h_x$$
$$\frac{\partial}{\partial x} \to h_y$$



Laplacian zero-crossing



Smoothing is usually introduced either before or after the filtering

Revisiting Line detection

- Possible filters to find gradients along vertical and horizontal directions

−1	−1	−1	−1	0	1
0	0	0	−1	0	1
1	1	1	−1	0	1

Prewitt

Averaging provides noise suppression

−1	−2	−1	−1	0	1
0	0	0	−2	0	2
1	2	1	−1	0	1

Sobel

This gives more importance to the center point.

Edge Detection

a b
c d

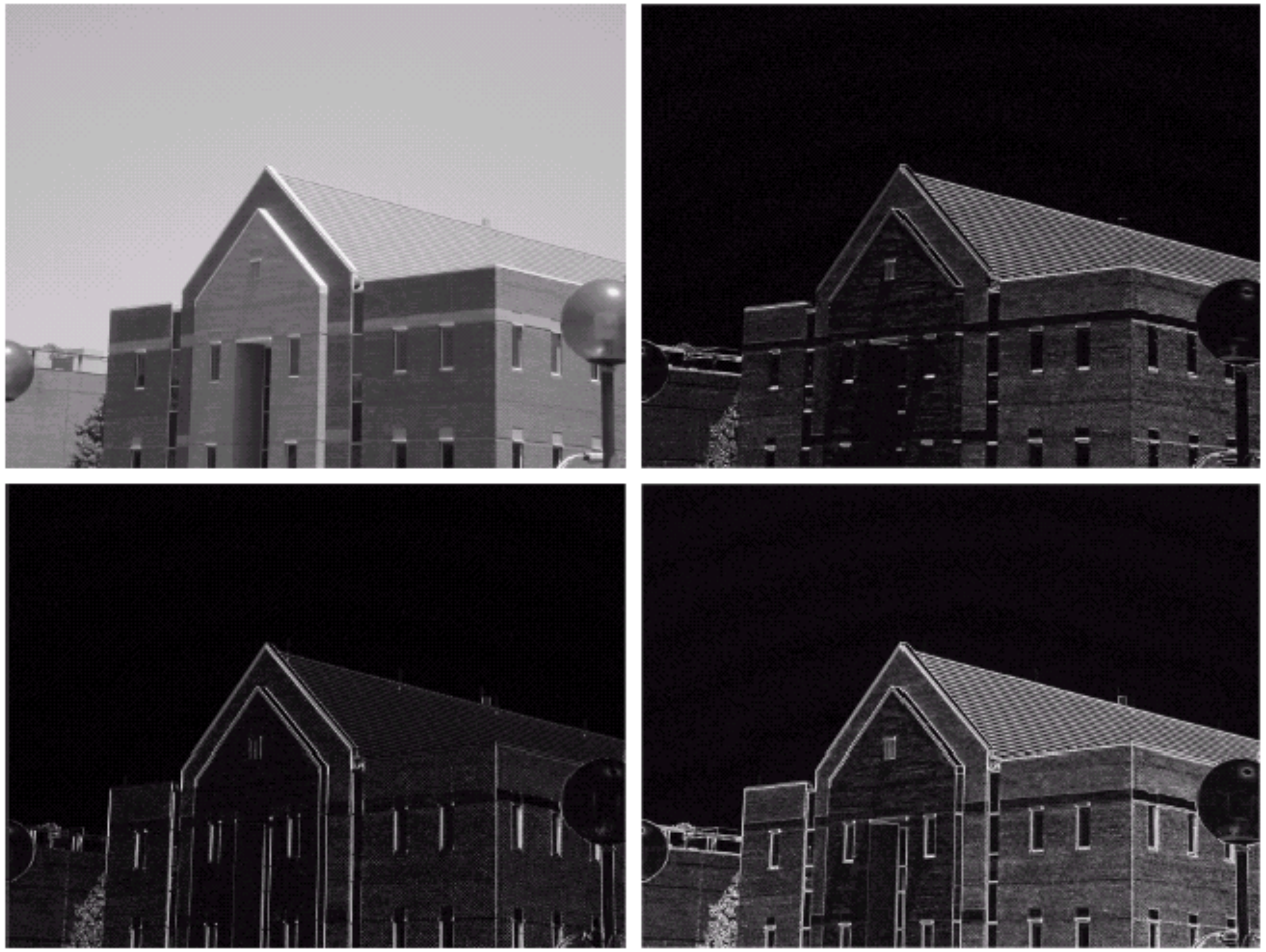


FIGURE 10.10 (a) Original image. (b) $|G_x|$, component of the gradient in the x-direction. (c) $|G_y|$, component in the y-direction. (d) Gradient image, $|G_x| + |G_y|$.

Edge Detection

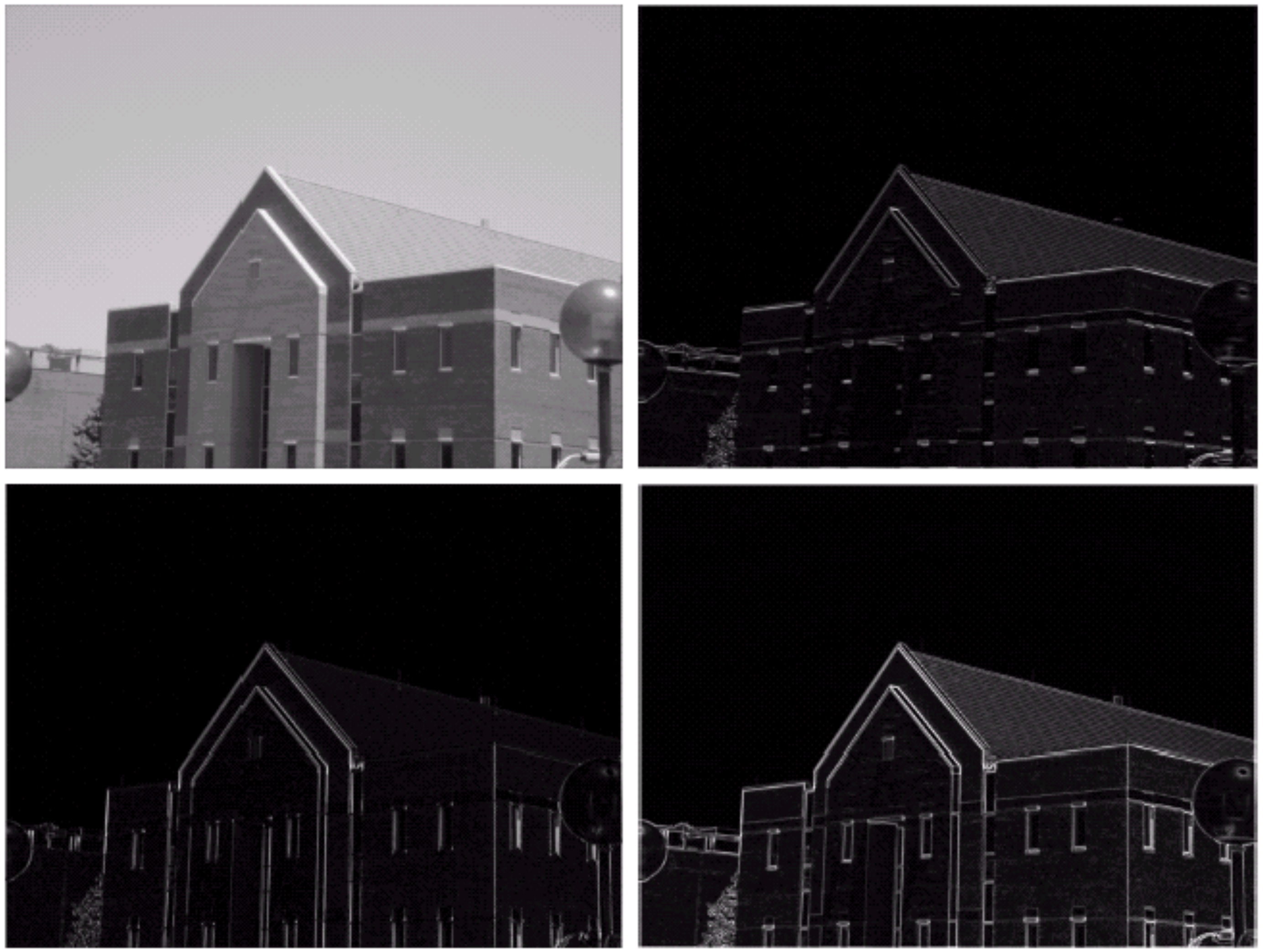
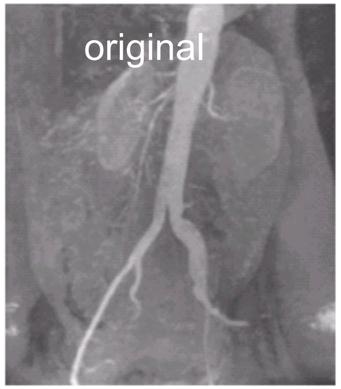
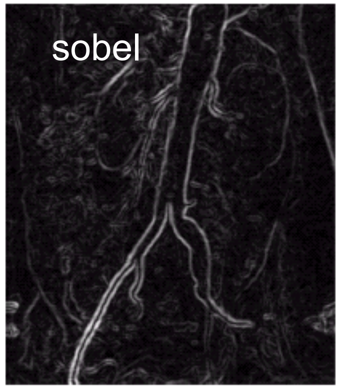


FIGURE 10.11 Same sequence as in Fig. 10.10, but with the original image smoothed with a 5 × 5 averaging filter.

Edge Detection

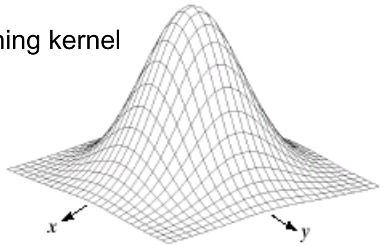




a b
c d
e f g

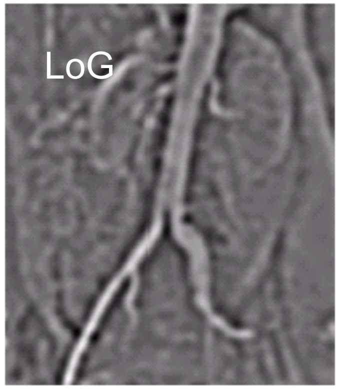
FIGURE 10.15 (a) Original image. (b) Sobel gradient (shown for comparison). (c) Spatial Gaussian smoothing function. (d) Laplacian mask. (e) LoG. (f) Thresholded LoG. (g) Zero crossings. (Original image courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)

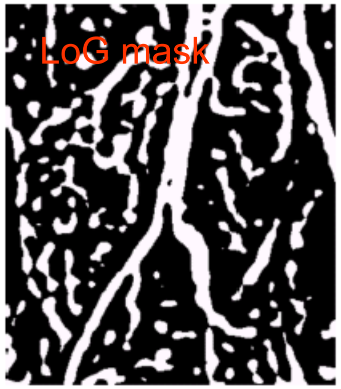
smoothing kernel

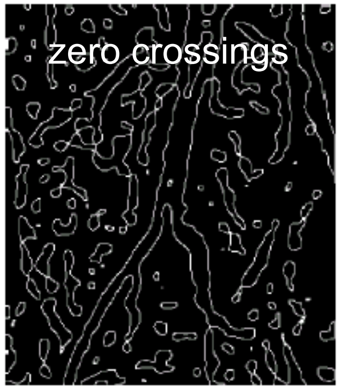


–1	–1	–1
–1	8	–1
–1	–1	–1

Laplacian mask





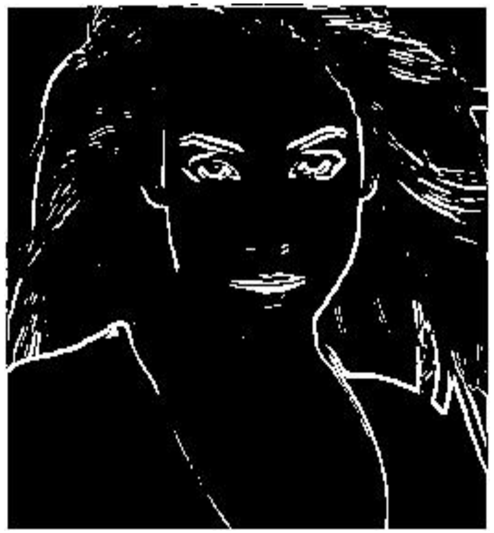


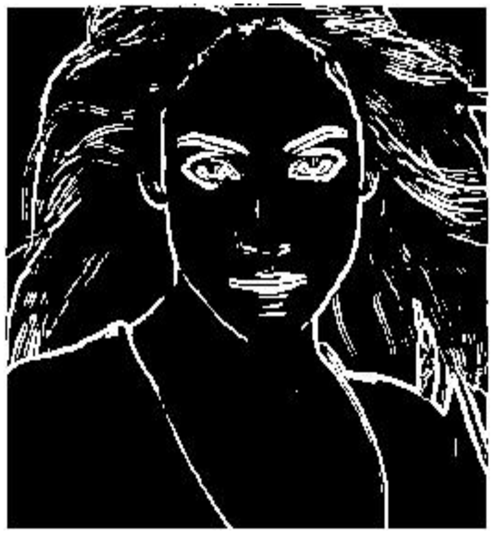
One simple method to find zero-crossings is black/white thresholding:
1. Set all positive values to white
2. Set all negative values to black
3. Determine the black/white transitions.

Compare (b) and (g):
- Edges in the zero-crossings image is thinner than the gradient edges.
- Edges determined by zero-crossings have formed many closed loops.

Edge detection: Gradient thresholding

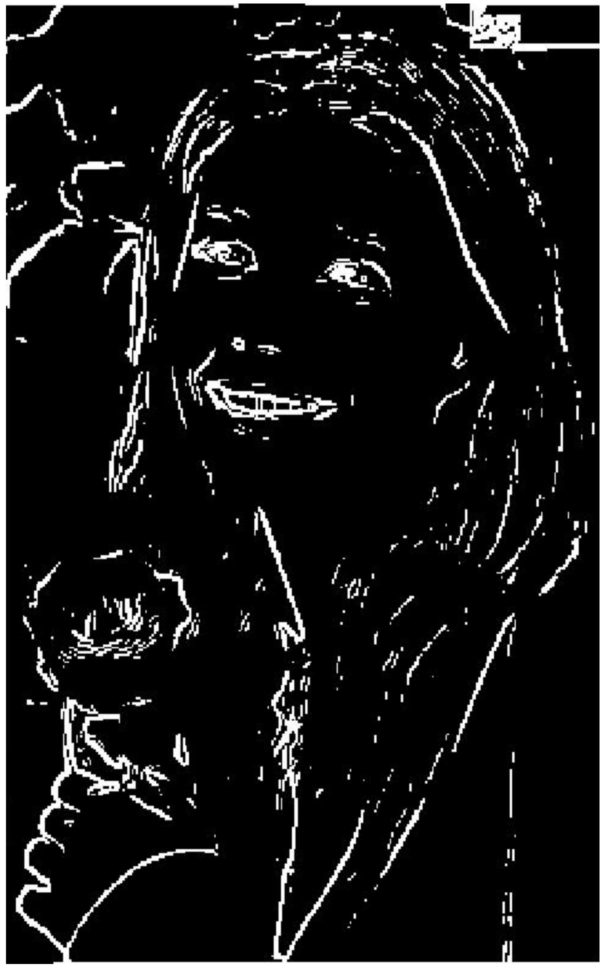
Prewitt filter: decreasing the threshold

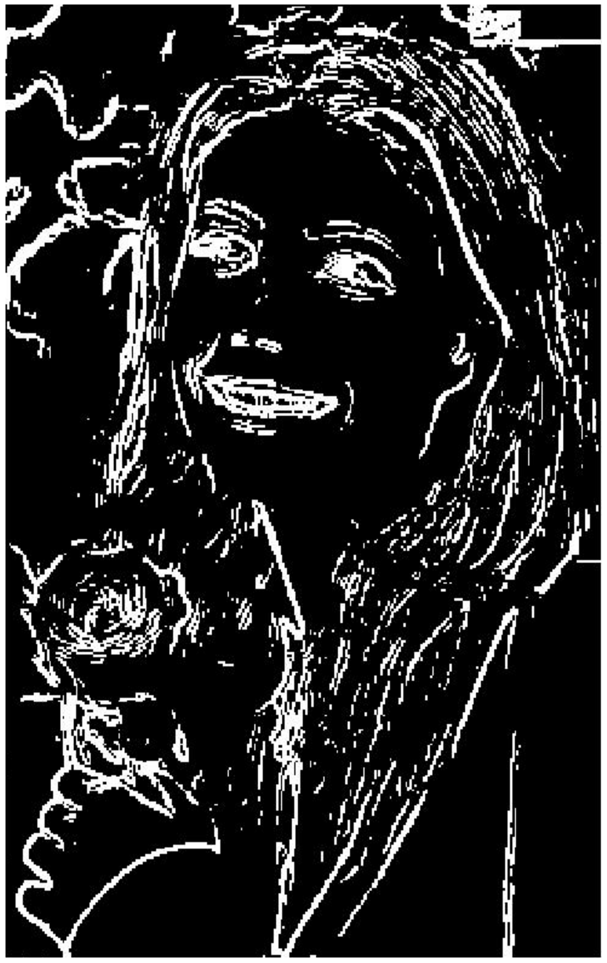




Edge detection: Gradient thresholding

Prewitt filter: decreasing the threshold

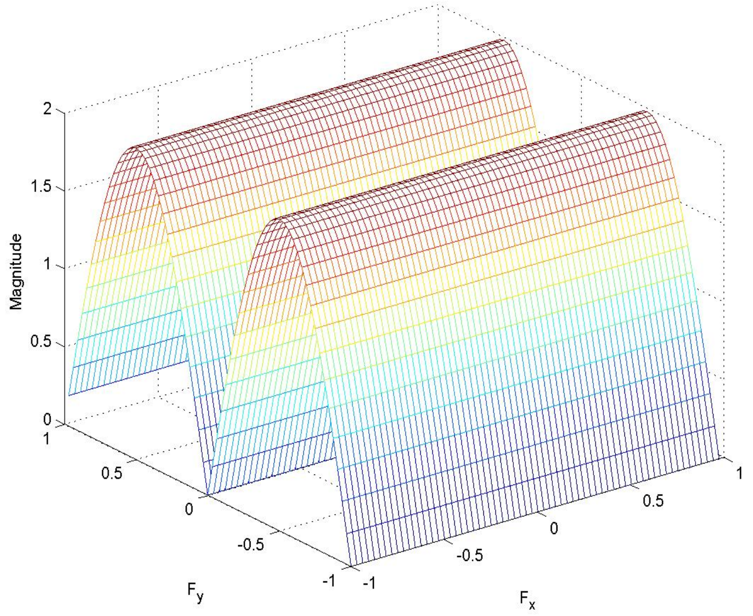


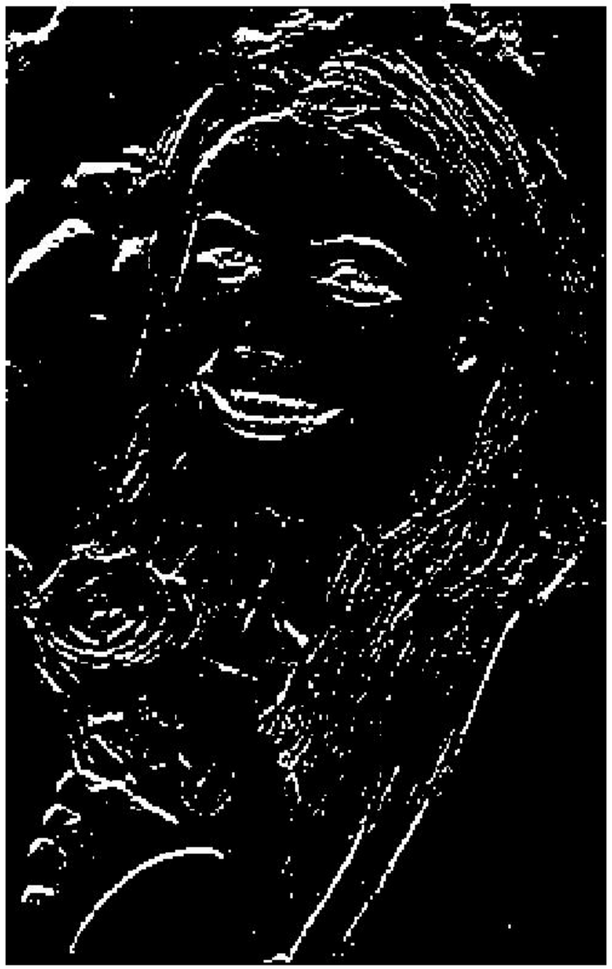


Edge detection

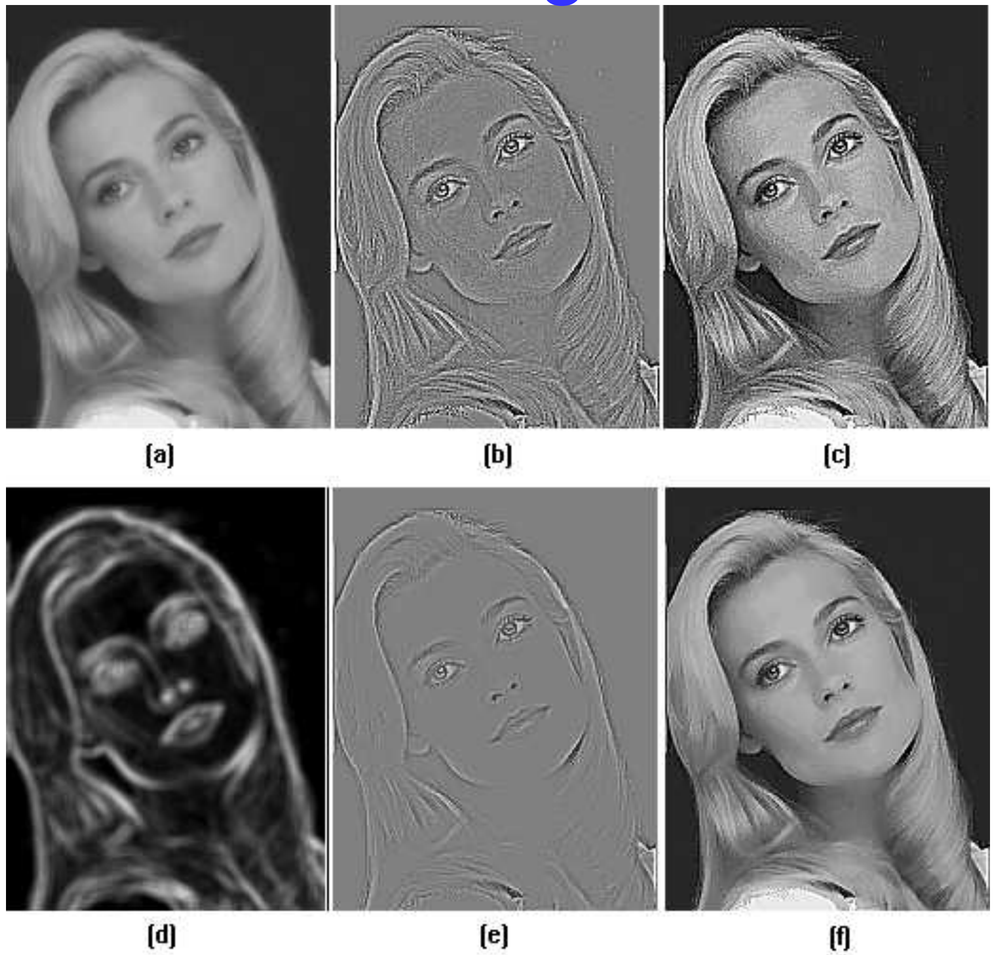
Using only the vertical high frequencies

$$h_{highpass} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$





Application to image enhancement



(a) Input image; **(b)** Laplacian of (a); **(c)** Spatially invariant high-pass filtering [sum of (a) and (b)]; **(d)** Mask image [Sobel gradient of (a) smoothed by a 5x5 box filter]; **(e)** Product of (b) and (d); **(f)** Space-variant enhancement [sum of (a) and (e)].

Multiscale edge detection

- The information obtained by filtering the image at different scales is combined to determine the edge map
  - scale ↔ width (s, sigma parameter) of the filter
- Different possibilities
  - Adapting the filter bandwidth to the local characteristics of the image (Wiener)
  - Combining edge maps obtained at different scales
- Canny algorithm
  - Smoothing (allows for different scales)
  - Gradient maxima
  - Two thresholds to detect both *weak* and *strong* edges. Weak edges are retained if they are connected to strong ones (labeling)
  - Less sensible to noise

Canny algorithm

- Based on a 1D continuous model of a step edge of amplitude h_E plus additive Gaussian noise of amplitude σ_n
- The impulse response of the filter h(x) is assumed to be FIR and *antisymmetric*
- *First and second order* derivatives are used
- Two thresholds have to be chosen
- Criterion: the Canny operator impulse response h(x) is chosen to satisfy three criteria
  - Good *detection*
  - Good *localization*
  - *Single response*

Canny

- Simultaneous goals:
  - low rate of detection errors
  - good edge localization
  - only a single detection response per edge.
- Canny assumed that false-positive and false-negative detection errors are *equally undesirable* and so gave them equal weight.
- Canny determined the optimal 1-D edge detector for the step edge and showed that its impulse response can be approximated fairly well by the derivative of a Gaussian

Canny

- The effect of the derivative of Gaussian is to prevent multiple responses by smoothing the truncated signum in order to permit only one response peak in the edge neighborhood.
- The choice of variance for the Gaussian kernel controls the filter width and the amount of smoothing.
  - This defines the width of the neighborhood in which only a single peak is to be allowed.
  - The variance selected should be proportional to the amount of noise present.
  - Because the edges in a given image are likely to differ in signal-to-noise ratio, a single-filter implementation is usually not best for detecting them. Hence, a thorough edge detection procedure should operate at *different scales*.

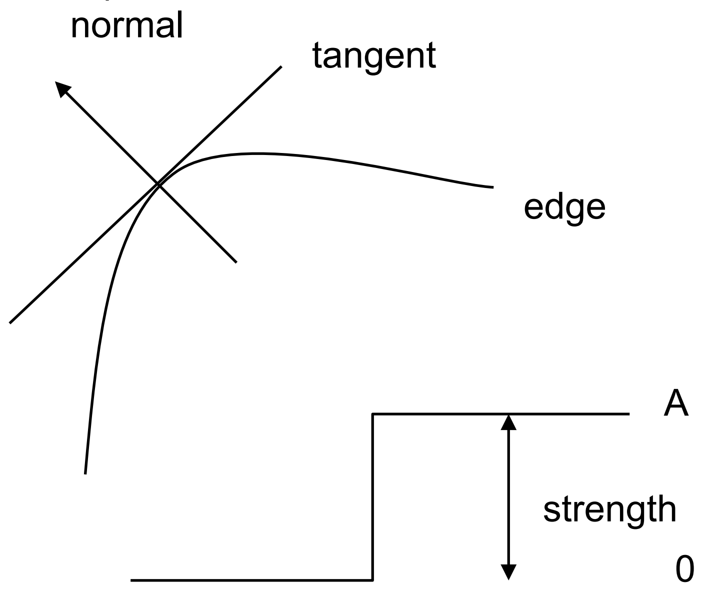
Canny algorithm

1. Apply Gaussian filter to smooth the image in order to remove the noise
2. Find the intensity gradients of the image
3. Apply non-maximum suppression to get rid of spurious response to edge detection
4. Apply double threshold to determine potential edges
5. Track edge by hysteresis: Finalize the detection of edges by suppressing all the other edges that are weak and not connected to strong edges

Step edge model

- Parameters
  - Edge direction (tangent to the curve)
  - Normal direction (vector orthogonal to the contour at edge location)
  - Local contrast (edge strength)
  - Edge location (along the normal direction)

$$G(x) = \begin{cases} 0, & x < 0 \\ A, & x \geq 0 \end{cases}$$



Step 2: find intensity gradients

- Canny algorithm uses four/six filters to detect horizontal, vertical and diagonal edges in the blurred image
- The modulus and direction of the gradient are calculated in the four images
- The edge direction angle is rounded to one of four angles representing vertical, horizontal and the two diagonals (0, 45, 90 and 135 degrees for example)

Step 2: about directional filters

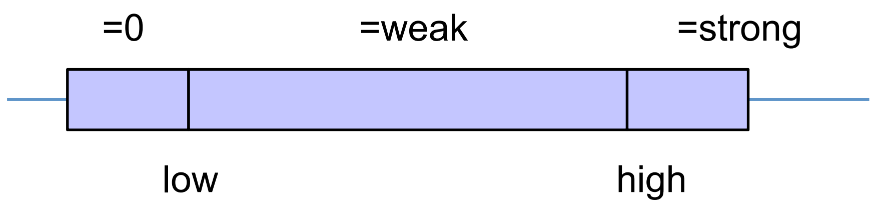
- Each mask includes a derivative of Gaussian function to perform the nearly optimal directional derivative across the intended edge.
  - A smooth, averaging profile appears in the mask along the intended edge direction in order to reduce noise without compromising the sharpness of the edge profile.
  - In the smoothing direction, the filter extent is usually several times that in the derivative direction when the filter is intended for straight edges.
- Canny’s method includes a “goodness of fit” test to determine if the selected filter is appropriate before it is applied.
  - The test examines the gray-level variance of the strip of pixels along the smoothing direction of the filter. If the variance is small, then the edge must be close to linear, and the filter is a good choice. A large variance indicates the presence of curvature or corner, in which case a better choice of filter would have smaller extent in the smoothing direction.

Step 3: non maximum suppression

- Edge thinning technique to sharpen the edges
- Compare the edge strength of the current pixel with the edge strength of the pixel in the positive and negative gradient directions
  - If the edge strength of the current pixel is the largest compared to the other pixels in the mask with the same direction (i.e, the pixel that is pointing in the y direction, it will be compared to the pixel above and below it in the vertical axis), the value will be preserved. Otherwise, the value will be suppressed

Step 4: double threshold

- Two thresholds
  - Two threshold values are set to clarify the different types of edge pixels, one is called *high threshold* value and the other is called the *low threshold* value.
  - If the edge pixel's gradient value is *higher* than the *high* threshold value, they are marked as *strong* edge pixels.
  - If the edge pixel's gradient value is *smaller* than the *high* threshold value and *larger* than the *low* threshold value, they are marked as *weak* edge pixels.
  - If the pixel value is *smaller* than the low threshold value, they will be *suppressed*.
  - The two threshold values are empirically determined values, which will need to be defined when applying to different images.

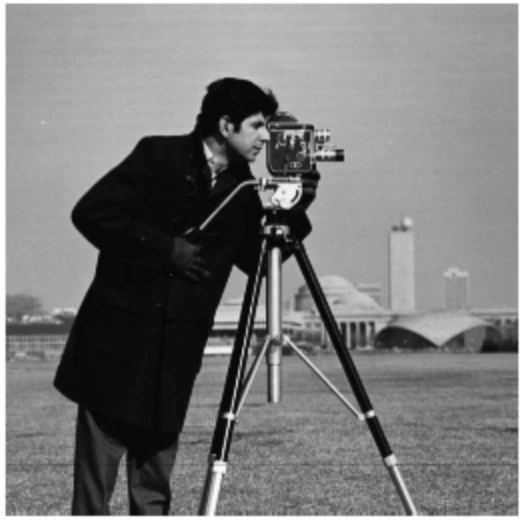


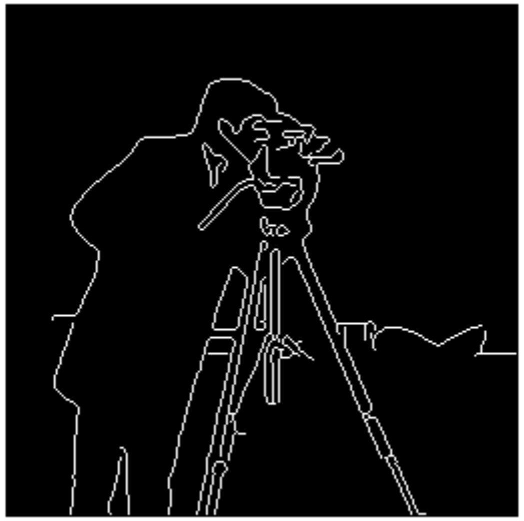
Step 5: track edge by histeresys

- Which weak edge pixels should we keep?
- Usually the weak edge pixel caused from true edges will be connected to the strong edge pixel.
- To track the edge connection, Binary Large Object-analysis is applied by looking at a weak edge pixel and its 8-connected neighborhood pixels.

➢ As long as there is one strong edge pixel is involved in the BLOB, that weak edge point can be identified as one that should be preserved.

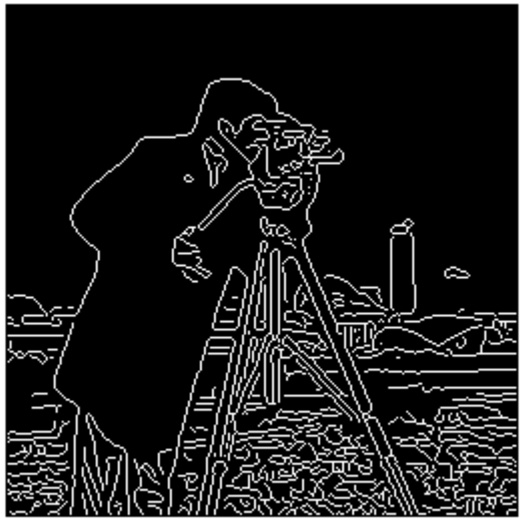
Example





Example

threshold = 0.5



Performance assessment

- Possible errors
  - False negatives (an edge point is present but it is not detected)
  - False positives (a non-edge point is detected)
  - Error in the estimation of the orientation
  - Error in the localization of the edge

- Paradigms
  - Use of synthetic images + noise with known parameters
  - Tests on sets of real images

Performance evaluation

Objective

- The **ground truth is assumed to be available** and represented by the actual contour (*full reference metric*)
- Concerns low level features
  - Measure to which extent the estimated contour represents the actual contour
- Metric: MSE among the estimated ($f[j,k]$) and the real ($s[j,k]$) edges

Subjective

- The ground truth is not necessarily given (*reduced or no-reference metric*)
- Concerns high-level features
  - Measures to which extent the estimated contour allows to identify the corresponding object in the image
  - Focus on semantics or image content
- Metric: subjective scores given to the different algorithms
- Lead to perception-based models and metrics

Objective assessment

- 1D case

estimated edge

- 2D case

$$E = \int_{x_0-L}^{x_0+L} \left[f(x) - S(x)\right]^2 dx$$

$$E = \iint \left[f(x,y) - s(x,y)\right]^2 dxdy \qquad (3)$$

ground truth

*A common strategy in signal detection theory is to establish **a bound on the probability of false detection** resulting from noise and then try to maximize the probability of true signal detection*

- When applied to edge detection, this translates in **setting a the *minimum value of the threshold*** such that the *FP rate does not exceed the predefined bound*. Then the probability of true edge detection can be calculated by a coincidence comparison of the edge maps of the ideal versus the real edge detectors

Performance assessment: Figure of Merit

- Types of errors
- Detection
  - Missing valid edge points (False Negatives, FN)
  - Failure to *localize* edge points
  - Classification of noise fluctuations as edge points (False Positives, FP)
- Localization
  - Error in estimating the edge angle;
    - Mean square distance of the edge estimate from the true edge
- Accuracy
  - Algorithm's tolerance to distorted edges and other features such as corners and junctions

Performance assessment: Figure of Merit

ensures a penalty for smeared or fragmented edges

$$F_M = \frac{1}{\max(I_A, I_I)} \sum_{i=1}^{I_A} \frac{1}{1 + d_i \alpha^2}$$

$$F_M = 1: \text{ perfectly detected edge}$$

to penalized edges that are localized by offset from the true position

I$_I$, *I*$_A$: number of ideal and detected edge points, respectively
d$_i$: distance among the ideal and the detected edge point along the normal to a line of ideal edge points (evaluated according to (3))
α: scaling constant
The rating factor is normalized so that R = 1 for a perfectly detected edge

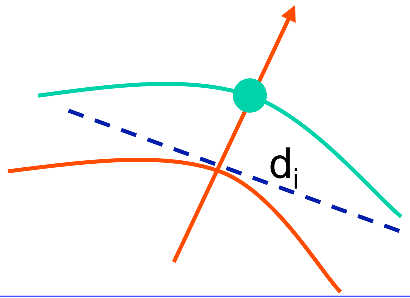


Figure of merit

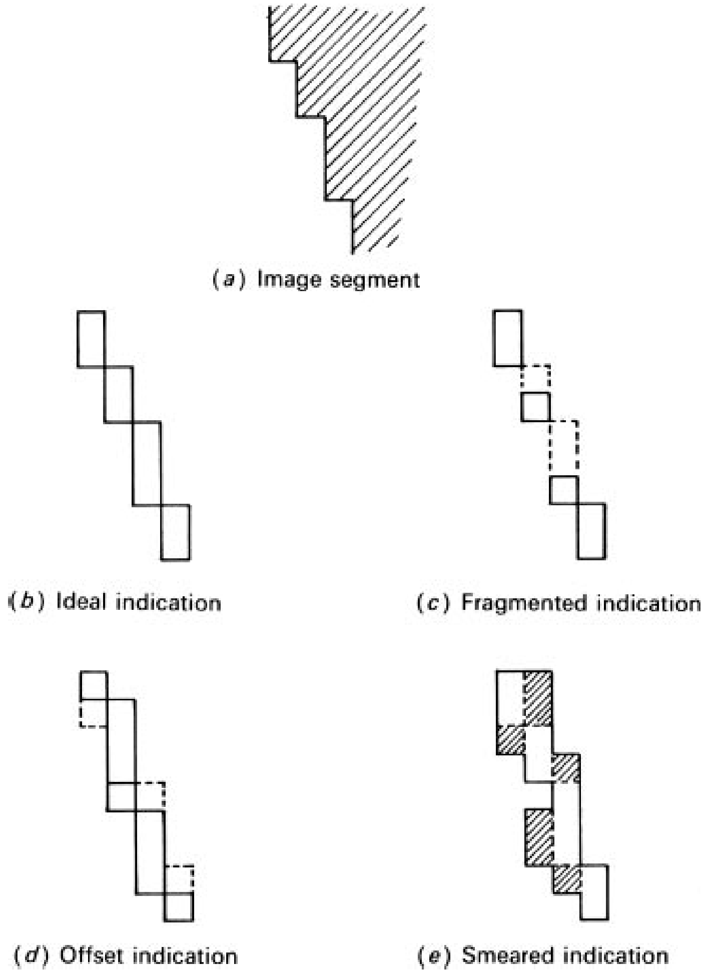
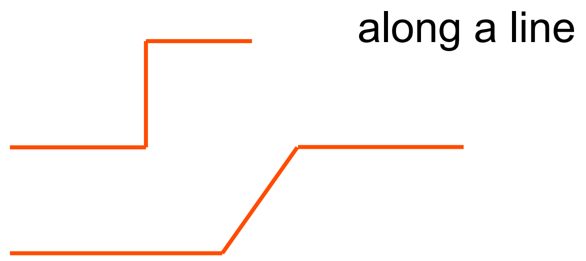


FIGURE 15.5-10. Indications of edge location.

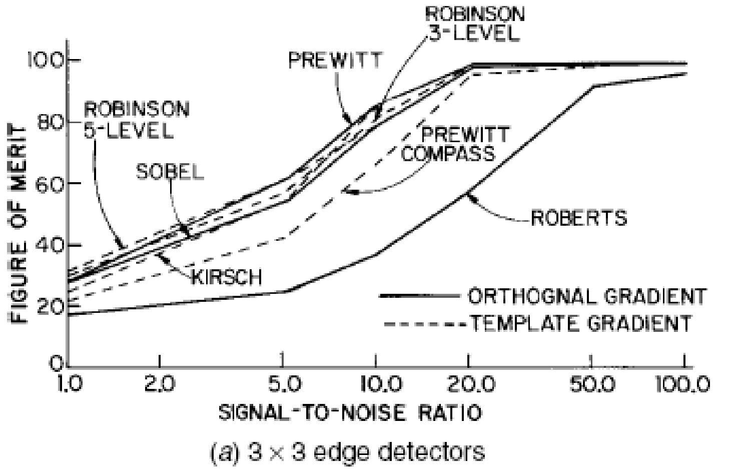
Filters competition



- A possible classification strategy
  - Synthetic image
    - 64x64 pixels
    - vertical oriented edge with variable slope and contrast
    - added Gaussian noise of variance σ_n
    - Control parameter SNR=(h/ σ_n), h being the normalized edge value (0<h<=1)
  - Filter threshold: maximize the FM constrained to maximum bound for false detection rate
    - False detection=false positives
    - Probability to detect an edge when no edge is present

Filter comparison

Step edge (w=1)



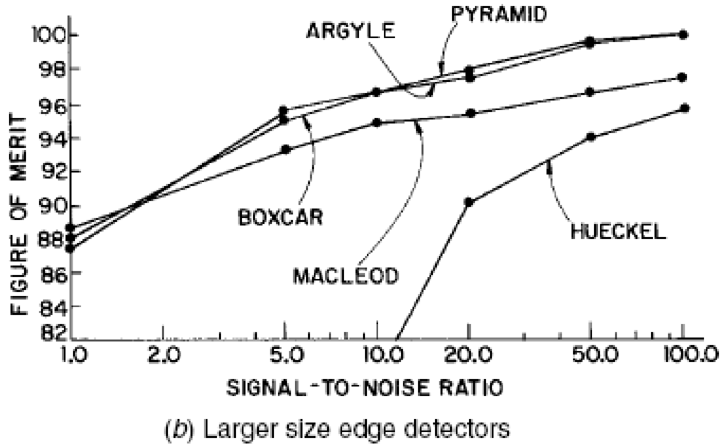


FIGURE 15.5-11. Edge location figure of merit for a vertical ramp edge as a function of signal-to-noise ratio for $h = 0.1$ and $w = 1$.

FOM is low for wide and noisy edges; and high in the opposite case

Filter comparison

Ramp edge

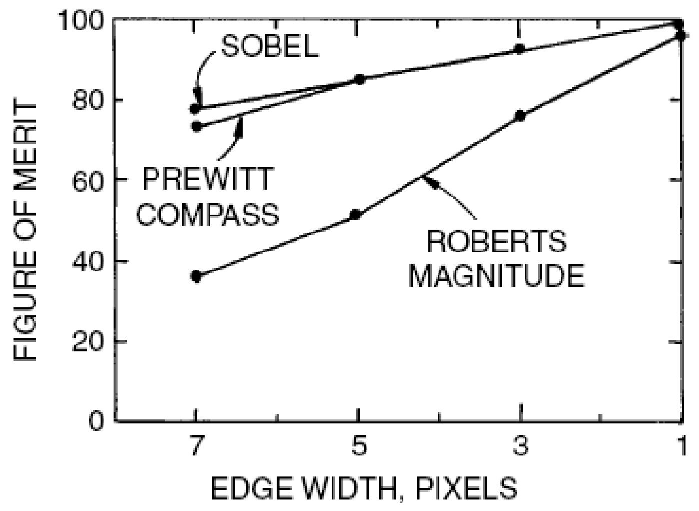


FIGURE 15.5-12. Edge location figure of merit for a vertical ramp edge as a function of signal-to-noise ratio for $h = 0.1$ and SNR = 100.

Changing SNR

- Sobel
- Step edge

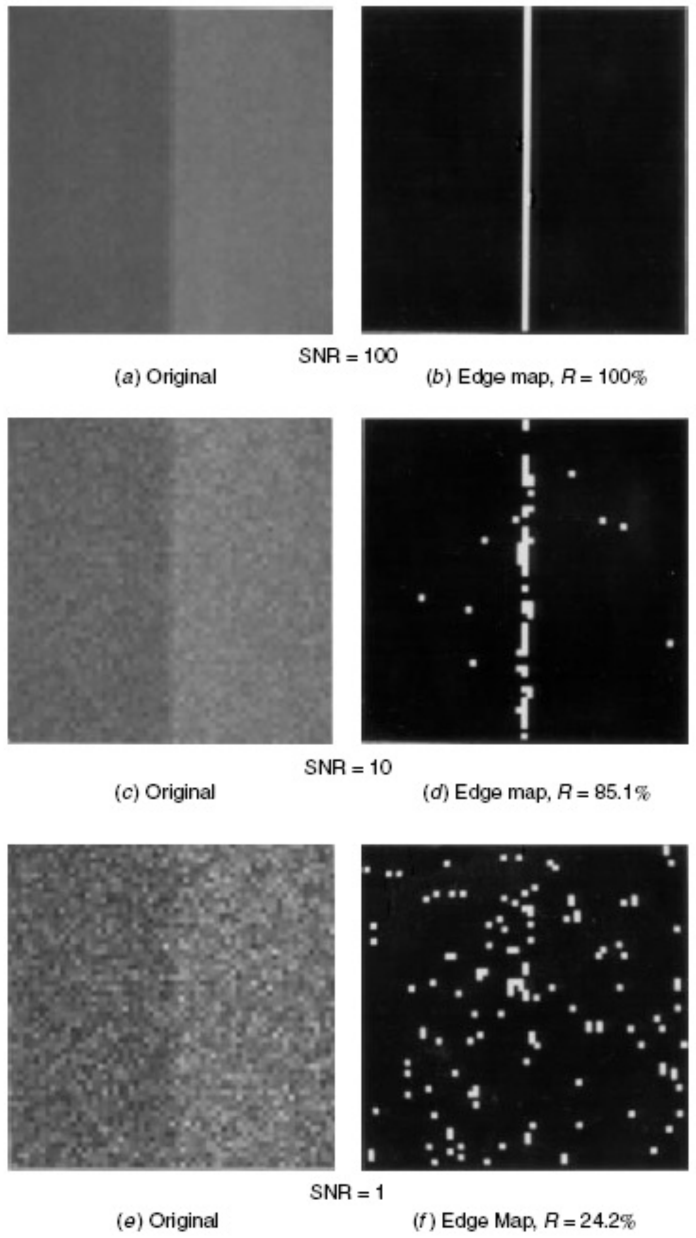

SNR = 100
(a) Original
(b) Edge map, $R = 100\%$
SNR = 10
(c) Original
(d) Edge map, $R = 85.1\%$
SNR = 1
(e) Original
(f) Edge Map, $R = 24.2\%$

FIGURE 15.5-13. Edge location performance of Sobel edge detector as a function of signal-to-noise ratio, $h = 0.1$, $w = 1$, $a = 1/9$.

Changing the filter

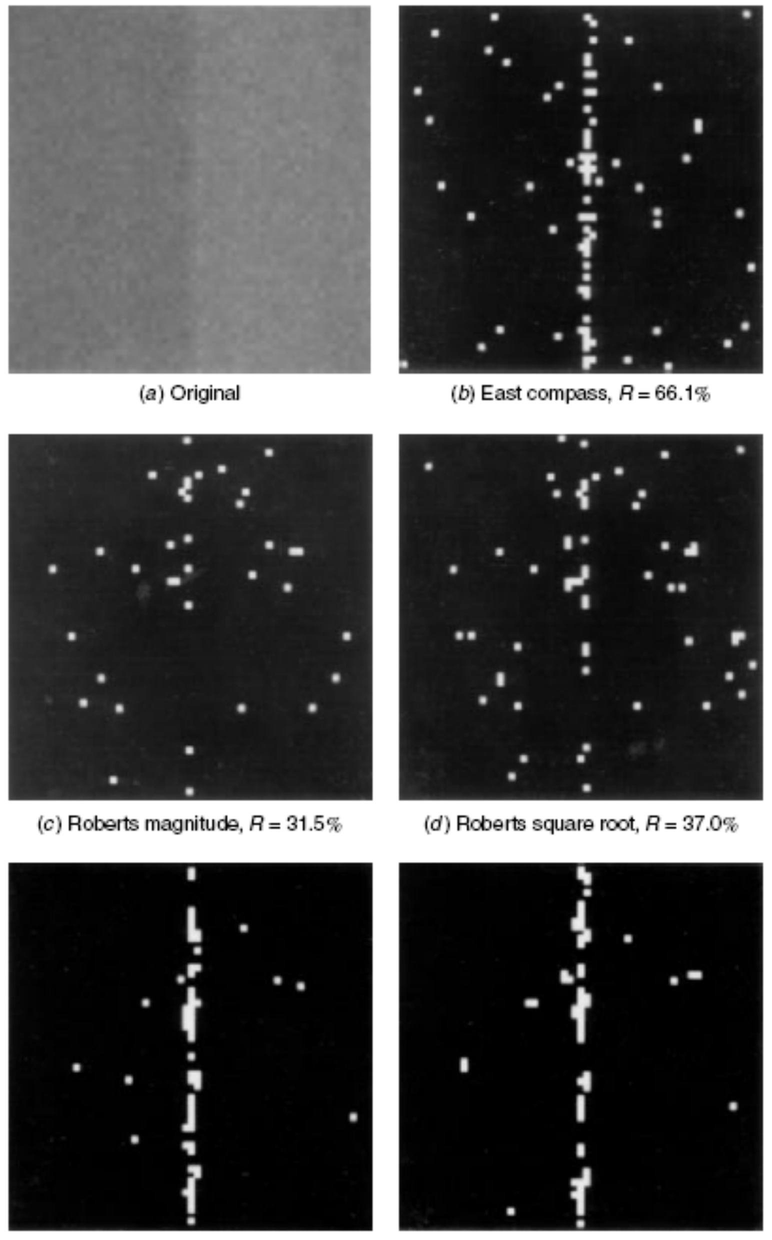
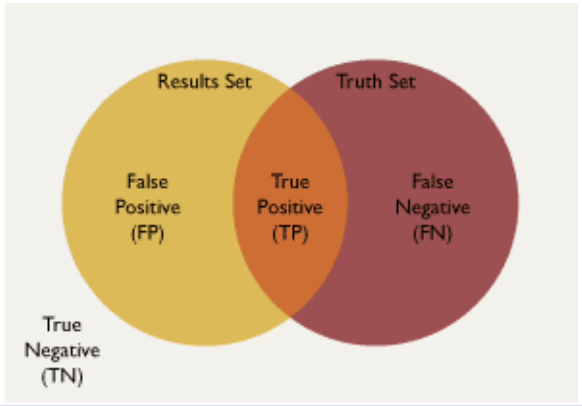


FIGURE 15.5-14. Edge location performance of several edge detectors for SNR = 10, $h = 0.1$, $w = 1$, $a = 1/9$.

Jaccard index

- The Jaccard similarity index, also known as the Tanimoto coefficient, measures the overlap of two sets.
- It is defined as the size of the intersection of the sets divided by the size of their union. In other words,
- $J(A, B) = |A \cap B| / |A \cup B|$
- where A and B are the two sets.
- This can also be expressed in terms of the true positive (TP), false positive (FP), and false negative (FN) sets as TP / (FP+TP+FN).



Jaccard index

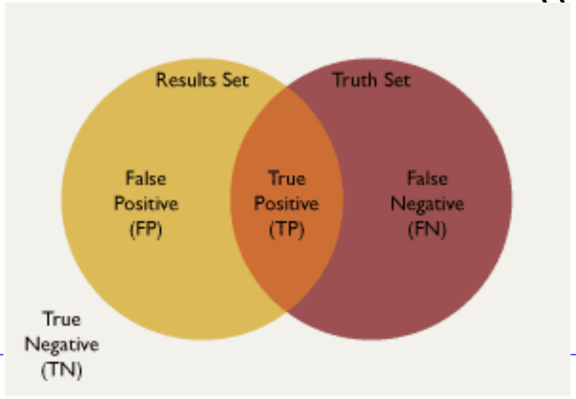
- The Jaccard index is zero if the two sets are disjoint, i.e., they have no common members, and is one if they are identical.
- Higher numbers indicate better agreement in the sets, so when we apply this index to evaluate the agreement of brain segmentation segmentation results, the goal is to get as close to 1 as possible.

Dice index

- Like the Jaccard similarity index, the Dice coefficient also measures set agreement.
- In this case, the measure is given by the formula:

D(A,B) = 2 |A and B | / (|A|+|B|)

- where A and B are the two sets. More simply, this formula represents the size of the union of 2 sets divided by the average size of the two sets.
- In terms of false positive, false negative, true negative, and true positive counts, these numbers may also be written as 2 TP/((FP +TP)+(TP +FN))



Dice index

- A value of 0 indicates no overlap; a value of 1 indicates perfect agreement.
- Higher numbers indicate better agreement, and in the case of segmentation indicate that the results match the gold standard (or the reference) better than results that produce lower Dice coefficients.

Subjective evaluation

- Task: "Give a score to the detected edges"
- Many trials
  - The experiment is repeated at least two times for each subject
- Many subjects
  - A sufficiently high number of subjects must participate in the experiment to make data analysis significant from the statistical point of view
- Output: {scores}
- Data analysis

A high figure of merit generally corresponds to a well-located edge upon visual scrutiny, and vice versa.